2017

Distributed optimization of multi-agent systems: Framework, local optimizer, and applications

Yue Zu
Iowa State University

Distributed optimization of multi-agent systems: Framework, local optimizer, and applications

by

Yue Zu

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Aerospace Engineering

Program of Study Committee:
Ran Dai, Co-major Professor
Ganesh Rajagopalan, Co-major Professor
Peng Wei
Yan-bin Jia
Jing Dong

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2017

DEDICATION

I would like to dedicate this thesis to my parents, Zhiying Zu and Guoli Ma. I could not successfully complete my Ph.D program and finishing the thesis writing without their continuous encouragement and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my appreciation to those who have ever helped me conducting my research and thesis writing. Firstly, my major advisor, Dr. Ran Dai, continuously supports my Ph.D study and research. Her patience, hard working and immense knowledge motivates me for completing my Ph.D program. Besides my advisor, I would also like to appreciate my committee members: Dr. Peng Wei, Dr. Ganesh Rajagopalan, Dr.Yan-bin Jia, and Dr. Jing Dong for their helping on the class, academic guidance on my research, and insightful comments on my thesis. Their efforts and concerns encourage me going further on my research road. My sincere thanks also go to the research group members, including Chuangchuang Sun, Changhuang Wan, Nathaniel Kingry, Yen-chen Liu and Logan Towers. Team collaboration expands my academic perspectives all the time.

ABSTRACT

Convex optimization problem can be solved in a centralized or distributed manner. Compared with centralized methods based on single-agent system, distributed algorithms rely on multi-agent systems with information exchanging among connected neighbors, which leads to great improvement on the system fault tolerance. Thus, a task within multi-agent system can be completed with presence of partial agent failures. By problem decomposition, a large-scale problem can be divided into a set of small-scale sub-problems that can be solved in sequence/parallel. Hence, the computational complexity is greatly reduced by distributed algorithm in multi-agent system. Moreover, distributed algorithm allows data collected and stored in a distributed fashion, which successfully overcomes the drawbacks of using multicast due to the bandwidth limitation. Distributed algorithm has been applied in solving a variety of real-world problems. Our research focuses on the framework and local optimizer design in practical engineering applications. In the first one, we propose a multi-sensor and multi-agent scheme for spatial motion estimation of a rigid body. Estimation performance is improved in terms of accuracy and convergence speed. Second, we develop a cyber-physical system and implement distributed computation devices to optimize the in-building evacuation path when hazard occurs. The proposed Bellman-Ford Dual-Subgradient path planning method relieves the congestion in corridor and the exit areas. At last, highway traffic flow is managed by adjusting speed limits to minimize the fuel consumption and travel time in the third project. Optimal control strategy is designed through both centralized and distributed algorithm based on convex problem formulation. Moreover, a hybrid control scheme is presented for highway network travel time minimization. Compared with no controlled case or conventional highway traffic control strategy, the proposed hybrid control strategy greatly reduces total travel time on test highway network.

CHAPTER 1. OVERVIEW

1.1 Introduction

1.1.1 Optimization Problems

Nowadays, engineers pay attention not only to the feasibility of system design that works at a certain level, but also the best design that improves a desired system performance [Parkinson et al. (2013)]. This motivation leads to the development and research of optimization algorithms. Optimization modeling and solution searching methods have been widely used in business management and finance [Cornuejols and Tütüncü (2006); Zenios (2002)], chemical [Bhaskar et al. (2000); Rangaiah (2016)], civil [Awad et al. (2012); Bejan and Lorente (2001)], electrical, and control engineering fields [Goldbergt (1992); Grefenstette (1986); Gaing (2004)], and etc.

To find the optimal design of a real-world system, we first formulate the optimization problem to mathematically describes the purposes and requirements of the system design. An optimization problem consists of an objective function and a set of constraints to be satisfied by the optimal design approach. According to work stated in Gershenfeld (1999), solving an optimization problem finds the optimal solution of unknown variable $\mathbf{x} \in \mathbb{R}^n$ that minimizes/maximizes the objective function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$. The solution is subject to a set of equality $h_i(\mathbf{x})$ and inequality constraints $g_i(\mathbf{x})$. A maximizing problem can be handled as a minimizing problem by assigning the negative objective function.

The form of a general optimization problem is expressed as follows, where constraints $g_i(\mathbf{x})$ and $h_i(\mathbf{x})$ are optional according to the real-world problem definition.

$$\begin{aligned} min. \quad & f(\mathbf{x}) \\ s.t. \quad & g_i(\mathbf{x}) \leq 0, i = 1, ..., m \\ & h_i(\mathbf{x}) = 0, i = 1, ..., p \end{aligned} \tag{1.1}$$

1.1.2 Convex and Non-Convex Optimization

Optimization problems are generally categorized into convex and non-convex problem in terms of nature of objective and constraints definition. Further classification is specified and shown in Fig.1.1. As a fundamental formulation, a convex optimization problem minimizes a convex objective function (or maximize a concave objective) over a convex set [Bertsekas and Scientific (2015)]. Engineers and researches gain many benefits from solving convex problem with fast convergence speed, global optimum, and the feasibility to be embedded in the real-time system design. On the other hand, a non-convex optimization problem is the one where the objective is non-convex, or the optimal solution is defined within a non-convex feasible region. Searching for the optimal solution of a non-convex optimization problem may be trapped at a local optimum and challenged by heavy demands of computational time. It generally take time in the exponential order to the amount of constraints and variables to determine feasibility of a problem. Even for a close-to-convex problem, the optimization can be computationally intractable [Boyd and Vandenberghe (2004)].

1.2 Research Motivation, Methodology, and Purpose

1.2.1 Motivation: Multi-Agent System

Benefit from mathematical modeling of convex optimization, our research focuses on convex problem formulation and efficient algorithm development. Typically, a convex problem can be solved in either a centralized manner by a single agent, or a decentralized fashion,

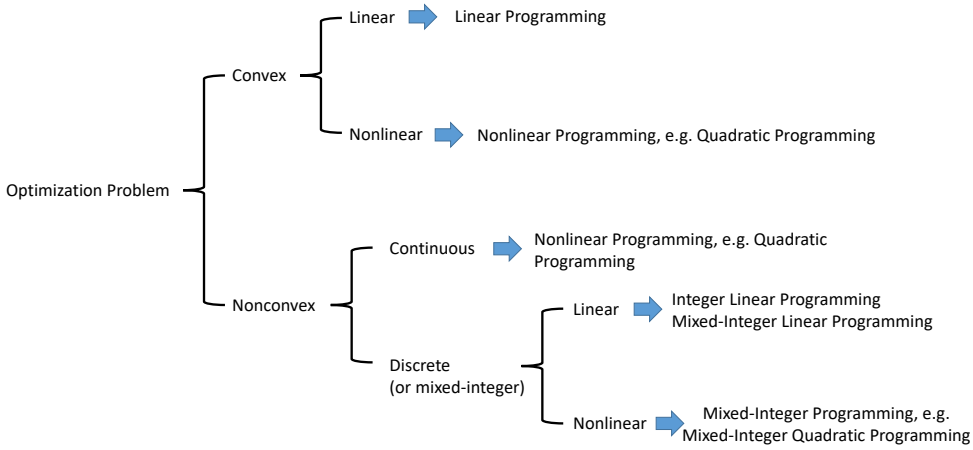

Figure 1.1 Classification of Optimization Problem. Blue arrows indicate the corresponding solving techniques for optimal solution.

a.k.a. distributed optimization method, by introducing multiple agents. For a large-scale convex optimization problem where there are high-dimensional variables, large number of constraints, and great amount of data, distributed optimization method could be one of the critical way to improve the scalability and reliability of the system design. In this case, the computational efficiency could not reduce too much, or even improve as dealing a huge data by a sequential/parallel computation grid. Solving a convex optimization problem by distributed methods in a multi-agents system leads to the following performance improvements compared with centralized methods based on a single-agent system.

Table 1.1 Comparison of Multi-Agents System with Distributed Optimization Method and Single-Agent System with Centralized Optimization Method

	Multi-Agent System	**Single-Agent System**
Data Collection Type	Globally Collecting	Locally Collecting
Scale of the Problem Solved in Each Agent	High Dimension (more constraints & variables)	Low Dimension (less constraints & variables)
Failure Tolerance	Single Point Failure	No Single Point Failure
Communication with Agent	Multicast (high cost)	Local Communication (low cost)

1.2.2 Methodology: Distributed Optimization Algorithm

Distributed optimization algorithms have attracted the attention of researchers and engineers for decades. Inspired by the well-developed algorithms for solving unconstrained optimization problems, Lagrangian relaxation are introduced to construct dual problems. By doing so, most of constraints can be eliminated in dual problems. Based on Lagrangian relaxation with dual variables, dual decomposition is proposed [Everett III (1963)]. This leads to one classical distributed algorithm that employs dual decomposition and the subgradient method [Wei et al. (2010)]. Similar to the gradient descent method for solving primal problem, subgradient is applied in maximizing dual sub-objective in an iterative manner. J. N. Tsitsiklis and other researchers investigated further on distributed asynchronous gradient-based optimization method [Tsitsiklis et al. (1986)] and parallel computation [Tsitsiklis (1984); Bertsekas and Tsitsiklis (1989)]. Recently, dual subgradient distributed method is widely applied in optimal control [Low and Lapsley (1999); Kelly et al. (1998)], Natural Language Processing [Rush et al. (2010); Sontag et al. (2012)], Markov Random Fields optimization [Komodakis et al. (2011, 2007)], finance problems [Schütz et al. (2009); Berkelaar et al. (2002)], and etc.

Based on dual decomposition and subgradient method, many calculus-based distributed algorithms have been proposed and applied in multi-agent system. Back to the 1970's, Daniel G., et al. introduced an augmented Lagrangian method [Gabay and Mercier (1976)]. An augmented Lagrangian incorporates a convex quadratic term in addition to ordinary Lagrangian, which equivalently formulates a strictly convex problem. From then on, further studies on augmented Lagrangian have been conducted. A simple but efficient distributed algorithm, Alternating Direction Method of Multiplier (ADMM) was established. Benefit from the dual decomposition, ADMM follows a distributed scheme to update primal and dual variables. Moreover, ADMM implements augmented Lagrangian by which the application is extended to solving a general convex optimization problem without the strict convexity assumption Boyd et al. (2011). At the same time, Accelerated Dual Descent (ADD)

method is another significant evolution for distributed algorithm. Michael Z. in his paper [Zargham et al. (2014); Zarghamy et al. (2013)] presented ADD for network optimization. Inspired by the fast convergence rate of Newton's method, Michael Z. and his co-authors proved the existence of a generalized dual Hessian. Instead of calculating dual Hessian using global information, they employed local information to compute a generalized dual Hessian which leads to a fast ascent direction for maximizing dual objective. Meanwhile, Ermin W., et al. developed the distributed Newton method [Wei et al. (2010, 2013a,b)] and applied the method to solve Network Utility Maximization (NUM) problem. Primal and dual iterations are presented for the Newton direction calculation in a distributed fashion.

Another popular distributed algorithm employs a consensus scheme to locally exchange information among a subsets of agents. Information aggregation is carried out through a network by using this non-calculus-based algorithm. Two typical approaches solving for consensus problem are Anti-entropy and Gossiping [Jelasity et al. (2005); Jelasity (2011); Boyd et al. (2005)]. It has been verified that the difference between the current state and the consensus state eliminates linearly with some assumptions associated with updating and network topology [Wei et al. (2010)]. To reach consensus on a scalar value or calculate the average of the initial value of agents, Gossiping or Anti-entroy is effective and efficient, but with a slow convergence speed [Nedic and Ozdaglar (2009)]. However, for a more general problem where additional non-consensus conditions are required, such consensus scheme may not be applicable.

1.2.3 Purpose and Achievements

Our research is on the basis of calculus-based algorithm, i.e. the gradient-based method. Distributed algorithm is applied either to dual sub-problem, or directly to primal sub-problem after decomposition. According to the specific requests in real-world engineering projects, we try to design efficient and powerful local optimizers by introducing multiple agents and associated distributed optimization algorithm. In the first project *Spacial Motion*

Estimation of 3D Rigid Body, spacial motion estimation is conducted by designing many local estimators. Dual subgradient and distributed Newton method are verified providing a higher estimation accuracy and less time spent than using Extended Kalman Filter (EKF) algorithm based on a single-agent-single-sensor system [Zu et al. (2014b,a)]. In the second project *Path Planning for In-Building Evacuation*, a distributed path planning scheme is proposed for in-building evacuation when hazard occurs. Distributed computation effectively overcomes the system single point failure due to the connection damage, [Zu and Dai (2017)]. In the third project *Intelligent Highway Traffic Management*, highway traffic management problem is converted to convex optimization problem by developing a simplified solution based on the first-order traffic flow model. This simplified solutions are successfully applied to minimizing fuel consumption and travel time. The highway traffic management problem is solved in both centralized and distributed fashion [Zu et al. (2016)]. The rest of this section describes the projects background and main contributions of the PhD thesis.

1.3 Applications

1.3.1 Spacial Motion Estimation of 3D Rigid Body

Spatial rigid motion with both translational and rotational evolutions has been found in a variety of dynamical systems, such as robotics, spacecraft, biomechanics, physics engines, and many others. Precise real-time motion estimation is critical for autonomous motion control when disturbances that requires frequent tracking of the motion states are considered. The commonly used motion tracking instrument is the Inertial Navigation System (INS) equipped with a computer, motion sensors, and rotation sensors to continuously provide position, orientation, and velocity of a moving object [Savage (2013); Groves (163)]. However, when such an INS instrument is not available or the concerned object is not accessible for INS installation, real-time estimation of both translational and rotational motion of a dynamical system becomes a challenging task.

One of the challenges comes from the coupled translational and rotational motion, which requires a compact and efficient mathematical model to describe the combined motion. Nonlinearity of expressions describing translational and rotational motion makes the real-time estimation extremely difficult, especially when Euler angles are included in the rotational expression. Quaternions have been introduced as mathematical tools for calculation involving three-dimensional (3D) rotations to avoid singularity and reduce expensive computational load created by Euler angle expressions [Hamilton (1844)]. As alternative and powerful tools for representing object orientation, quaternions have been acknowledged as playing indispensable role in dynamical systems due to their unambiguous, unencumbered, and computationally-efficient features [Dai and Sun (2015)]. However, when both rotations and translations simultaneously occur in dynamical systems, quaternions alone cannot represent the spatial transformations. Thus, dual quaternions, an extension of quaternions, have been invented to unify the representation of rotational and translation motion within a single invariant coordinate frame. In many practical applications, such as robot arms and satellite attitude control, dual quaternions have demonstrated their advantages in terms of compactness, non-singularity, and computational efficiency [Filipe and Tsiotras (2014); Aspragathos and Dimitros (1998); Wu et al. (2005); Wang et al. (2013); Filipe and Tsiotras (2013); Filipe et al. (2015)]. In this project, the dual quaternion kinematics is introduced to represent spatial rigid motion. The dual quaternion based representation avoids using trigonometric functions, which leads to a non-singular representation and also simplifies its kinematics model as differentiating polynomials with respect to quaternion elements is simpler than differentiating trigonometric functions with respect to Euler angles.

Existing work relating to rigid motion estimation using dual quaternion models applied an EKF to estimate translational and rotational motion using a single image sensor [Goddard and Abidi (1998); Olsson et al. (2003); Chiang et al. (2008); Lin et al. (2010)]. However, if traditional image-based sensors are used for motion tracking, a portion of the sensing data will not be available when the tracking points or lines move out of its viewing zone or when

the sensor vision is blocked by interference. Such missing measurement information due to visual constraints or sensor malfunction creates difficulty in maintaining continuity and completeness of the tracking data. Furthermore, the complicated motion, the huge amount of observation data, and the resulting computational burden motivates the investigation of an efficient estimation algorithm for processing the observed data in real time.

Considering the challenging observational environment and the constraints of individual sensors, a distributed estimation scheme, in which the operation of each sensor is independent of others while cooperation among sensors is allowed, is more applicable to spatial rigid motion estimation of complex dynamical systems. The computation burden could be relived by solving each subproblem with unchanging dimensions. Distributed estimation technology has been commonly used in process control, signal processing and information systems [Mesbahi and Egerstedt (ps 8); Acikmese et al. (2008); Peterson and Paley (2013); Sun and Xin (2015a)]. A subset of these efforts have generally been focused on the integration of measurements from all the sensors into a common estimate without using a centralized processor. For example, literature in [Olfati-Saber (2009)] contributes a great deal of work towards achieving an average consensus among distributed filters. By implementing a low-pass or band-pass filter, the estimates will reach an average consensus in a distributed manner. Continuous efforts relative to such types of work have been applied by extension to heterogeneous network systems and continuous Kalman filters, producing fast convergence of a decentralized consensus [Sun and Xin (2015b); Xiao and Boyd (2004); Schizas et al. (2008); Cattivelli and Sayed (2010); Garin and Schenato (2010)]. In this project, we propose to solve the distributed estimation problem by formulating it as a multi-agent optimization problem. We hence firstly propose a distributed estimation scheme via dual decomposition methodology and subgradient method, named as dual subgradient method, to estimate the space object motion using multiple sensors, where the kinematics of the object is based on the dual quaternion model. After that, inspired by the rapid convergence of the Newton's method in solving network utility maximization problems [Dolev et al. (2009); Wei et al.

(2010)], we propose to design a Newton-type distributed estimation algorithm aimed at increasing the rate of convergence.

The main contribution is to design a multi-sensor framework for estimation of the spatial rigid motion based on dual quaternions. Two types of distributed algorithms are generated by formulating the multi-sensor estimation problem as a multi-agent optimization problem with linearly coupled constraints. We verify that Newton-type distributed estimation algorithm has faster convergent rate than the one from dual subgradient algorithm when processing the multi-sensor measurement data in a connected network.

1.3.2 Path Planning for Building Evacuation

In emergency situations, it is extremely important for all of the occupants to evacuate from dangerous regions through safe paths in public buildings, such as schools, malls, and supermarkets. Efficient evacuation strategies will significantly reduce casualties in natural or man-made disasters, e.g., hurricane, earthquake, fire, and terrorist attack. Continuous efforts on improving techniques related to evacuation, including sensor networks [Filippoupolitis et al. (2008); Barnes et al. (2007)], information exchanging [Gorbil et al. (2011); Shklovski et al. (2008)], and building design [Sagun et al. (2014); Lui et al. (2015)], have been initiated to protect evacuees during evacuation. The focus of these techniques is to develop specific approaches based on precise modeling of occupants movement and hazard spreading modes to obtain an optimal evacuation path regarding to a desired evacuation performance, for instance, minimum evacuation time.

Existing movement models are classified into two categories, namely macroscopic and microscopic models [Pelechano and Malkawi (2008)]. Macroscopic models pay attention to the location, arrangement, and schedule of the evacuee flow among rooms in public buildings. Without considering interaction among evacuees, macroscopic model formulates the evacuation management and schedule problem as a network flow optimization problem [Choi et al. (1988); Mamada et al. (2004); Tjandra (2003)]. Rooms and available exits are handled

as nodes or vertices and each feasible path between two nodes is considered as an edge in a graph. Moreover, each edge has an assigned cost representing a desired evaluation index, such as the distance or time required for evacuees to transit along the relative edge. Based on the macroscopic model, the formulated network flow optimization can be solved via existing optimization algorithms, such as linear programming [Hamacher and Tjandra (2001); Tan et al. (2011); Yao et al. (2009)], genetic algorithm [Adeli and Cheng (1994); Liu et al. (2006); Li et al. (2010); Cheng et al. (2008)], and simulated annealing algorithm [Jahangiri et al. (2011)]. When hazard spreading model is considered in a dynamic evacuation graph, the edge cost will be updated due to the dynamically changing hazard areas [Tabirca et al. (2009); Barnes et al. (2007)].

Microscopic models focus on individual social behaviors [Li and Xu (2014)]. For example, Pelechano and Malkawi [Filippidis et al. (2006)] consider psychological and physiological activities when modeling human movement. Additionally, they notice that communication among evacuees is a non-negligible social behavior when building a high precision model. Another example in Filippidis et al. (2006) considers the evacuees' interaction with the signage system, where a visibility catchment area has been developed and a prototype behavior model has been built for supermarket environments. In addition to the behavior-based model described above, there are many other types of microscopic model that have been widely used in evacuation planning, guidance and exit decision strategies. For example, work in Mesmer and Bloebaum (2014) proposes a novel egress decision model for exit choices based on utility and game theory. Microscopic behaviors, such as risk preferences, velocity determination and following others, are considered in the egress decision model. Interaction of evacuees has also been considered in cellular automata model which divides a large space into several cells where movement of evacuees in each cell is affected by evacuees in adjacent cells [Zhao et al. (2008); Psakhie et al. (2001); Miyamoto and Sasaki (1997); Wolfram (1983)].

However, most existing work searching for an efficient evacuation path employs a centralized sensing and computational framework and effects of congestion along the evacuation

path segments and around exits have not been considered. Due to spreading of hazard areas, the centralized computation node may become available, which leads to failure of generating evacuation guidance decisions. On the other facet, congestions have significant effect on evacuation time. In the worst case, jammed flow in vicinity of available exits may generate panic among evacuees. Thus a designed safe path based on the dynamic graph may not be available in real scenario if the congestion effects have not been considered.

This work utilizes a Cyber-Physical System (CPS) with networked sensing, information sharing, and distributed computation capabilities to generate optimal evacuation paths for scattered evacuees in a public building. The first step builds a dynamic evacuation graph with time-varying edge cost determined by both the hazard spreading model and decisions from each evacuee group. In addition, social behaviors are integrated into the graph-based macroscopic model by assuming that evacuees prefer to follow others in a group rather than move alone. Next, a Bellman-Ford-Dual-Subgradient (BFDS) algorithm is developed to search for the optimal evacuation paths while satisfying capacity constraints of corridors and exits in a distributed manner. Instead of computing evacuation paths for all groups in a centralized mode by collecting information, e.g., location and number of evacuees, from the entire graph, the distributed computational framework significantly reduces the scale and complexity of the original optimization problem. Meanwhile, the capacity constraints of corridors and exits are satisfied by coordination among individual groups to avoid congestion along the planned paths and around exits. The goal is to reduce the overall evacuation time for all evacuee groups by cooperative planning through CPS.

1.3.3 Intelligent Highway Traffic Management

Large scale complex transportation system is one of the indispensable infrastructures in urban and rural areas. The dramatically increasing demands of transportation service leads to traffic congestion, energy wasting and pollution, as well as safety issues. To deal with these issues, intelligent traffic management strategies relying on advanced sensing,

communication, and high performance computation techniques are attracting researchers' attention. Recent work in areas of intelligent transportation systems mostly focuses on modeling and reducing travel time [Daganzo (1995); Lu et al. (2008)] or minimizing delay at signalized intersections [Sims and Dobinson (1980); Guler et al. (2014); Li et al. (2016)]. If fuel consumption is considered in evaluating the transportation system performance, it is necessary to analyze the effectiveness of current traffic control systems in terms of energy efficiency while guaranteeing the accomplishment of transportation tasks in desired time.

Existing traffic control strategies are categorized into two application areas, i.e., urban roads and freeways. For traffic control of urban roads, developed work mainly focuses on signal-timing optimization. For example, a signal control system, named RHODES, aims to improve throughput and reduce the delay [Mirchandani and Head (2001)]. Another example employs the ant colony optimization algorithm to solve large scale traffic network problems [Putha et al. (2012)]. In areas of freeway traffic control, typical approaches include ramp metering control, such as ALINEA [Papageorgiou et al. (1991)] and METALINE [Messner and Papageorgiou (1990)], and dynamic speed limits control, e.g. the SPECIALIST proposed in Hegyi et al. (2008) for shockwave elimination. Furthermore, some of these work combine ramp metering and dynamic speed limits control to generate hybrid control commands. For example, in order to prevent traffic breakdown and relieve congestion, work in Hegyi et al. (2005) presents a predictive control approach for coordination of both ramp metering and dynamic speed limits.

An energy-efficient transportation system aims to reduce fuel consumption and emissions, e.g. CO, NO, CH_4, through eco-driving guidance. Existing eco-driving strategies for individual driving guidance focuses on training drivers behaviors, i.e., smooth acceleration, maintaining steady speeds, avoiding too fast speed, and etc., which has been verified to improve fuel economy on the order of 5-20 percentage [Barkenbus (2010)]. However, changing drivers' behaviors is a long-term effort and static driving advices may not guarantee prominent effects in dynamic traffic environments. Instead, recent studies concentrate on

traffic control and management strategies. For example, work in Liu et al. (2017) uses model predictive control (MPC) for traffic network based on a multi-class macroscopic traffic flow and emission model. Incorporated with end-point penalties, total time spent and emissions are further reduced. Another MPC-based method descibes an efficient en-route diversion strategy for real-time traffic flow control in [Luo et al. (2016)]. More energy-efficient traffic control approaches can be found in [Pasquale et al. (2015); Jamshidnejad et al. (2017); Han et al. (2016)], where the authors present nonlinear optimal control and gradient-based method in a MPC framework. However, although macroscopic traffic flow model, e.g. FASTLANE and METANET, have been adopted in energy-efficient traffic management, it is time consuming to find a convergent solution when a highly nonlinear traffic flow model is considered [Zegeye (2011)]. Speed intervals have been used to obtain an approximate solution without solving highly nonlinear dynamics, which results in accumulative errors over time [Dai et al. (2015)].

This work focuses on managing dynamic speed limit signs to control traffic flow speeds in order to reduce total fuel consumption during a specific time period. We adopt Lighthill-Whitham-Richard (LWR) macroscopic traffic flow model, introduced by Ligthill and Whitham in the 1950's (Lighthill and Whitham, 1955), and COPERT fuel consumption estimation model [Ntziachristos et al. (2000)]. Inspired by Barron-Jensen/Frankowska (B-J/F) solution for Hamilton-Jacobi (HJ) partial differential equations (PDEs) [Barron and Jensen (1990)], we use B-J/F solution to Moskowitz HJ PDEs to obtain exact solutions without approximation [Mazaré et al. (2011)]. It generates an explicit expression of solution based on a pre-specified fundamental diagram associated with initial and boundary conditions [Greenshields et al. (1935); Claudel and Bayen (2010a)]. Those analytical solutions are handled as model constraints incorporated in the optimization problem formulation. Furthermore, the solutions to Moskowitz HJ PDEs are simplified based on roadway decomposition and traffic status. Combing the simplified solution to Mozkowitz HJ PDEs with the quadratic formulation of COPERT, we formulate the energy-efficient traffic control problem as a convex

quadratic optimization problem (CQOP). The convex nature of the problem formulation guarantees convergence to a global optimal solution within polynomial computational time, which makes the approach feasible for real-time traffic control.

By using simplified solution to describe traffic dynamics, we also formulate a Linear Programming (LP) problem for travel time minimization. Furthermore, test highway is extended to a network scenario. Incorporating simplified solution, a mixed-integer quadratic programming problem with quadratic constraints (MIQQ) problem is constructed and a real-time traffic control strategy is proposed based on hybrid highway infrastructures.

The contribution include the following aspects. (1) Different from previous work that adopt triangular fundamental diagram for the derivation of explicit solution to Moskowitz HJ PDEs, new solution espressions are developed and simplified based on a parabolic shaped fundamental diagram associated with initial and boundary conditions. (2) By incorporating simplified solutions in model constraints, we formulate CQOP, LP and MIQQ for energy-efficient problem, travel time minimization problem in highway section and highway network, respectively. They are embedded in a real-time traffic management scheme to efficiently search for optimal commands. (3) Beyond the theoretical development in earlier work [Yue Zu and Dong (2016)], we implement CQOP, LP and MIQQ in real-world scenarios through VISSIM simulation by constructing a Component Object Model (COM) interface to connect MATLAB generated control commands with VISSIM simulation environments.

CHAPTER 2. CONVEX OPTIMIZATION PROBLEM AND CALCULUS-BASED CENTRALIZED OPTIMIZATION ALGORITHM

2.1 Introduction

Basically, optimization problem can be categorized into convex and non-convex problems. A convex optimization problem is characterized by convex functions, such as $f(\mathbf{x})$, $g_i(\mathbf{x})$ mentioned above and affinity for $h_i(\mathbf{x})$. The specific definition of convex optimization problem is described in Section 2.2. As a special case of optimization problem, convex optimization has been widely applied in many areas. The main advantage of formulating or modeling a real-world problem as a convex optimization problem is the problem solvability and solution reliability. Typically, there are many well-developed numerical algorithms, e.g. simplex method [Murty (1983)], interior point [Mehrotra (1992)], trust-region-reflective [Conn et al. (2000)], etc. that can efficiently solve convex optimization problem in a centralized way.

To derive a distributed algorithm, it is necessary to address classical calculus-based centralized method for solving unconstrained convex optimization problem. We introduce two typical algorithm: gradient descent and Newton's method in Sections 2.3 and 2.4.

As a first-order iterative algorithm, the gradient descent method searches for the solution in next iteration along negative direction of gradient at current solution. The algorithm always obtains a decreased objective value as iteration continues. For a strictly convex optimization problem, gradient descent method leads to single global optimum. As a simple but powerful algorithm, gradient descent is prevalent in finance, statistics, computer science, and etc. One of the most popular application is in machine learning. Gradient descent method

is implemented for learning large scale parameters. For example, the back-propagation for training neural network [Nielsen (2015)]. The detailed algorithm and convergence analysis are discussed in Section 2.3.

As a second-order iterative algorithm, the Newton's method tries to find the root of the differentiable objective, i.e. $f'(\mathbf{x}) = 0$. It takes the second-order information, i.e. the Hessian matrix into account. Hence, the Newton's method requires to compute the twice-differentiable objective function. Comparing to the gradient method, the Newton's method improves the convergence rate to a quadratic one. However, the drawback is the Hessian matrix calculation increases the computation complexity. When handling an unconstrained convex problem, the gradient method is the default choice in most cases. The optimizer performance in terms of objective value and convergence speed is used as the evaluation criterion. If it fails to fulfill the pre-specified request, some advanced algorithms, e.g. Newton's method, could be employed. We introduce Newton's method and the corresponding convergence analysis in Section 2.4.

2.2 Convex Set, Convex Function and Convex Optimization Problem

A convex set $\mathcal{C}$ corresponds to a convex region where for every pair of points in this region, the line segment connecting the two points is also within this region [Morris and Stark (2015)]. Mathematically, $\mathcal{C}$ is a convex set, if and only if $(1-t)\mathbf{x}_i + t\mathbf{x}_j$ is in $\mathcal{C}$, given any $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C}$ and $t \in [0, 1]$.

By defining a convex set $\mathcal{C}$, a function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is referred to as a convex function, if

$$f(t\mathbf{x}_i + (1-t)\mathbf{x}_j) \leq tf(\mathbf{x}_i) + (1-t)f(\mathbf{x}_j),\ \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{C}, t \in [0, 1]. \tag{2.1}$$

In other words, the epigraph is a convex set. Strong convexity holds if $t \in (0, 1)$ and Eq.(2.1) are satisfied. Now we can express the convex optimization problem in a general form of

$$\begin{aligned} min. \quad & f(\mathbf{x}) \\ s.t. \quad & g_i(\mathbf{x}) \leq 0, i = 1, ..., m \\ & A\mathbf{x} = b. \end{aligned} \tag{2.2}$$

Compared with standard form of optimization problem in (1.1), convex optimization problem is a special case which has more restrictions on objective function $f(\mathbf{x})$, inequality constraint $g_i(\mathbf{x})$ and inequality constraint $h_i(\mathbf{x})$. These restrictions include:

- $f(\mathbf{x})$ is a convex function.
- $g_i(\mathbf{x})$ is a convex function for all $i = 1, ..., m$. It formulates a convex feasible region.
- $h_i(\mathbf{x})$ is an affine function for all $i = 1, ..., p$. Therefore, we represent it as a compact form using matrix $A \in \mathbb{R}^{p \times n}$ and vector $b \in \mathbb{R}^p$.

For a concave maximization problem, we can simply handle it as a convex problem by minimizing the negative of its original objective. Furthermore, there is at most one optimum point if the objective is strong convex [Boyd and Vandenberghe (2004)].

2.3 First Order Method: Gradient Descent

2.3.1 The Algorithm

Gradient descent, a.k.a. steepest descent, is an iterative method which searches the state vector that monotonically decreases the objective value. Searching is conducted along the direction of negative gradient at current point $\mathbf{x}$. This direction allows fast reduction of the objective function if $f(\mathbf{x})$ is differentiable in a neighborhood of current point $\mathbf{x}$. Gradient descent method is summarized as

Initialization: $\mathbf{x}$ *in domain of* $f(\mathbf{x})$, $q = 0$

while *stopping criterion is not satisfied* **do**

Calculate gradient $\nabla f(\mathbf{x}^q)$ at current point $\mathbf{x}^q$.

Calculate and update state point by $\mathbf{x} = \mathbf{x} - \alpha^q \nabla f(\mathbf{x}^q)$.

end

Protocol 1 Gradient Descent Method

By linear search to determine a proper step size α^q, it guarantees that the objective value keeps decreasing after each state updating, i.e. $f(\mathbf{x}^{q+1}) \leq f(\mathbf{x}^q)$ where q is the iteration index. Therefore, $\mathbf{x}^q$ converges to local minimum as $q \to \infty$. Particularly, it converges to the global minimum for a convex optimization problem.

2.3.2 Convergence Analysis

We assume the Hessian of a strictly objective function $f(\mathbf{x})$ satisfies $mI \preceq \nabla^2 f(\mathbf{x}) \preceq MI$, m and M are two real positive numbers. Moreover, define the function $F(\alpha) = f(\mathbf{x}') = f(\mathbf{x} - \alpha \nabla f(\mathbf{x}))$.

We first consider a lower bound of a second-order Taylor approximation at point $\mathbf{x}$ [Boyd and Vandenberghe (2004)]:

$$\begin{aligned} f(\mathbf{x}') &\approx f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \frac{1}{2}(\mathbf{x}' - \mathbf{x})^T \nabla^2 f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) \\ &\geq f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \frac{m}{2}||(\mathbf{x}' - \mathbf{x})||_2^2 \\ &\geq f(\mathbf{x}) - \frac{1}{2m}||\nabla f(\mathbf{x})||_2^2. \end{aligned} \tag{3.3}$$

The second inequality holds due to $\nabla^2 f(\mathbf{x}) \succeq mI$. We take derivative of the right side of the second line and obtain the minimum value at $\mathbf{x}' = \mathbf{x} - \frac{1}{m}\nabla f(\mathbf{x})$, which obviously lead to the third inequality.

By the following derivation, we notice that an upper bound of this approximated function can be obtained.

$$\begin{aligned} f(\mathbf{x}') &\approx f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \frac{1}{2}(\mathbf{x}' - \mathbf{x})^T \nabla^2 f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) \\ &\leq f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \frac{M}{2}||(\mathbf{x}' - \mathbf{x})||_2^2 \\ &= f(\mathbf{x}) - \alpha||\nabla f(\mathbf{x})||_2^2 + \frac{M\alpha^2}{2}||\nabla f(\mathbf{x})||_2^2. \\ &\leq f(\mathbf{x}) - \frac{1}{2M}||\nabla f(\mathbf{x})||_2^2. \end{aligned} \tag{3.4}$$

In the above derivation, the second inequality holds due to $\nabla^2 f(\mathbf{x}) \preceq mI$. Then we replace $\mathbf{x}'$ by the state updating rule $\mathbf{x}' = \mathbf{x} - \alpha\nabla f(\mathbf{x})$ to obtain the third equality in (3.4). Moreover, the minimum value of $f(\mathbf{x}) - \alpha||\nabla f(\mathbf{x})||_2^2 + \frac{M\alpha^2}{2}||\nabla f(\mathbf{x})||_2^2$ is obtained at $\alpha = \frac{1}{M}$ by using the first order optimality condition on $F(\alpha) = f(\mathbf{x}')$. Hence the last inequality holds in (3.4).

We apply the result of (3.3) to the optimal solution $\mathbf{x}^*$ to give

$$f^*(\mathbf{x}^*) \geq f(\mathbf{x}) - \frac{1}{2m}||\nabla f(\mathbf{x})||_2^2. \tag{3.5}$$

Obviously, one has

$$||\nabla f(\mathbf{x})||_2^2 \geq 2mf^*(\mathbf{x}^*)f(\mathbf{x}). \tag{3.6}$$

By subtracting $f^*(\mathbf{x}^*)$ on both sides of (3.4) and applying (3.6), we have

$$\begin{aligned} f(\mathbf{x} - \alpha\nabla f(\mathbf{x})) - f^*(\mathbf{x}^*) &\leq f(\mathbf{x}) - f^*(\mathbf{x}^*) - \frac{1}{2M}||\nabla f(\mathbf{x})||_2^2. \\ &\leq (1 - \frac{m}{M})[f(\mathbf{x}) - f^*(\mathbf{x}^*)]. \end{aligned} \tag{3.7}$$

We add iteration index q in the state updating steps, i.e. $\mathbf{x}^q = \mathbf{x}^{q-1} - \alpha\nabla f(\mathbf{x}^{q-1}), ..., \mathbf{x}^1 = \mathbf{x}^0 - \alpha\nabla f(\mathbf{x}^0)$. We then repeatedly employ inequality result of (3.7) to get

$$f(\mathbf{x}^q) - f^*(\mathbf{x}^*) \leq (1 - \frac{m}{M})[f(\mathbf{x}^{q-1}) - f^*(\mathbf{x}^*)] \leq \cdots \leq (1 - \frac{m}{M})^q[f(\mathbf{x}^0) - f^*(\mathbf{x}^*)]. \tag{3.8}$$

Since $1 - \frac{m}{M} < 1$, we conclude the objective value converges to optimum $f^*(\mathbf{x}^*)$ as $q \to \infty$. In the following expressions, we found $\mathbf{x}^q$ converges linearly to the optimal solution $\mathbf{x}^*$.

$$\frac{f(\mathbf{x}^q) - f^*(\mathbf{x}^*)}{f(\mathbf{x}^{q-1}) - f^*(\mathbf{x}^*)} = 1 - \frac{m}{M} \tag{3.9}$$

2.4 Second Order Method: Newton's Method

2.4.1 The Algorithm

As a second order iterative optimization method for unconstrained optimization problem, the Newton's method employs the second order Taylor polynomial $f(\mathbf{x}+\Delta\mathbf{x})$ to approximate the original objective in the neighbor of $\mathbf{x}$. The second order approximation is written as

$$f_a(\mathbf{x}+\Delta\mathbf{x}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(\mathbf{x})\Delta. \tag{4.10}$$

By taking the derivative with respect to $\Delta\mathbf{x}$ and making it equal to zero, we have

$$\nabla f_a(\mathbf{x}+\Delta\mathbf{x}) = \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x})\Delta\mathbf{x} = 0. \tag{4.11}$$

We find the best step $\Delta\mathbf{x} = -\nabla^2 f(\mathbf{x})^{-1}\nabla f(\mathbf{x})$ which is referred to as Newton step. At current point $\mathbf{x}$, the Newton's method leads to the minimum of the approximated quadratic function f_a by taking the Newton step with an appropriate step size. For a state sequence $\mathbf{x}^q$, the Newton's method is summarized as

---

Initialization: $\mathbf{x}$ *in domain of* $f(\mathbf{x})$, $q = 0$.

while *stopping criterion is not satisfied* **do**

Calculate gradient $\nabla f(\mathbf{x}^q)$ at current point $\mathbf{x}^q$.

Calculate Hessian $\nabla^2 f(\mathbf{x}^q)$ at current point $\mathbf{x}^q$.

$\Delta\mathbf{x}^q = -\nabla^2 f(\mathbf{x}^q)^{-1}\nabla f(\mathbf{x}^q)$

Calculate and update state point by $\mathbf{x}^{q+1} = \mathbf{x}^q + \alpha^q \Delta\mathbf{x}^q$.

$q = q + 1$

end

Protocol 2 Newton's Method

The step size in the above algorithm is defined by $\alpha^q \in (0, 1)$. The Newton's method follows the similar methodology as in the gradient descent method, i.e. recursively searching

along the descent direction. Different from the gradient descent method, the second order approximation is implemented at current point $\mathbf{x}^q$. Minimizing this approximated function could be a very good optimizer of the original objective f if f is nearly quadratic and twice differentiable. Typically, the approximation perfectly describes the original objective f if f is a quadratic function. In this case, the Newton's method results in a one-step optimization. One can prove that the Newton's method has a quadratic convergence rate which is much faster than using gradient method.

2.4.2 Convergence Analysis

The objective function $f(\mathbf{x})$ is assumed to be strictly convex and twice differentiable. Similar to the proof provided in Section 2.3.2, the Hessian $\nabla f(\mathbf{x})$ satisfies $mI \preceq \nabla^2 f(\mathbf{x}) \preceq MI$, where m and M are two real positive numbers. The Hessian is Lipschitz continuous for $\mathbf{x}, \mathbf{y}$ on the domain of f with constant L. The Lipschitz condition on Hessian is stated as

$$||\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})||_2 \leq L||\mathbf{x} - \mathbf{y}||_2. \tag{4.12}$$

The Lipschitz condition is applied to prove the quadratic convergence rate as follows

$$\begin{aligned}
||\nabla f(\mathbf{x} + \alpha\Delta\mathbf{x})||_2 &= ||\nabla f(\mathbf{x} + \alpha\Delta\mathbf{x}) - \nabla f(\mathbf{x}) - \nabla^2 f(\mathbf{x})\alpha\Delta\mathbf{x}||_2 \\
&= ||\int_0^1 (\nabla^2 f(\mathbf{x} + \alpha\Delta\mathbf{x}) - \nabla^2 f(\mathbf{x}))\Delta\mathbf{x} d\alpha||_2 \\
&\leq \int_0^1 ||(\nabla^2 f(\mathbf{x} + \alpha\Delta\mathbf{x}) - \nabla^2 f(\mathbf{x}))\Delta\mathbf{x}||_2 d\alpha \\
&\leq \int_0^1 L\alpha||\Delta\mathbf{x}||_2^2 d\alpha \\
&= \frac{L}{2}||\Delta\mathbf{x}||_2^2 \\
&= \frac{L}{2}||\nabla^2 f(\mathbf{x})^{-1}\nabla f(\mathbf{x})||_2^2 \\
&\leq \frac{L}{2m^2}||\nabla f(\mathbf{x})||_2^2.
\end{aligned} \tag{4.13}$$

The first equality in Eq.(4.13) follows the result of Eq.(4.11). The fourth inequality holds by applying the Lipschitz condition inside the integral. At last, as it holds for $\nabla^2 f(\mathbf{x})^{-1} \preceq \frac{1}{m} I$, we obtain the last inequality in Eq.(4.13).

Multiplying $\frac{L}{2m^2}$ on both sides of the inequality result in (4.13), we repeatedly apply the result of (4.13) until it satisfies the stopping criterion. Then the following inequalities hold for $||\nabla f(\mathbf{x}^q)||_2 < n$,

$$\begin{aligned} \frac{L}{2m^2}||\nabla f(\mathbf{x}^q)||_2 &\leq (\frac{L}{2m^2}||\nabla f(\mathbf{x}^{q-1})||_2)^2 \\ &\leq (\frac{L}{2m^2}||\nabla f(\mathbf{x}^0)||_2)^{2^q} \\ &\leq \frac{1}{2}^{2^q} \end{aligned} \tag{4.14}$$

where $0 < n \leq \frac{m^2}{L}$. We then substitute the result of (4.14) into inequality (3.5) to give

$$\begin{aligned} f(\mathbf{x}^q) - f^*(\mathbf{x}^*) &\leq \frac{1}{2m}||\nabla f(\mathbf{x}^q)||_2^2 \\ &\leq \frac{1}{2m}\frac{4m^4}{L^2}(\frac{1}{2})^{2^{q+1}} \\ &= \frac{2m^3}{L^2}(\frac{1}{2})^{2^{q+1}}. \end{aligned} \tag{4.15}$$

Equation (4.15) demonstrates a rapid convergence to the optimal solution $\mathbf{x}^*$ when $q \to \infty$. This result shows a quadratic convergence rate using the Newton's method. Intuitively, the maximum difference of two objective values at current point and optimal point is reduces by $\frac{1}{2}^{2^q}$ after each iteration q.

2.5 Conclusion

In this section, we introduces the general form of optimization problem. An optimization problem is to minimize/maximize the objective by designing the optimal solution from all feasible solutions. Convex optimization problem is one of the important categories, which consists of convex objective function, convex feasible region, and a set of affine constraints. Formulating a real-world problem as convex optimization problem could be beneficial providing the efficient optimal searching algorithms and guaranteed global optimal solution.

Since constraints can be eliminated by relaxation, unconstrained optimization algorithms form the foundation of solving constrained optimization problems. Two typical calculus-based methods are discussed, including algorithm procedures and convergence analysis. First, gradient descent method only relies on the gradient information and line search algorithm. The computation is simple at each iteration. However, the primal solution is guaranteed to be located in a small neighbor of the optimal solution with a slow convergence rate, i.e. linear convergence. As an improved method, the Newton's method calculates the second-order information, i.e. the Hessian, at each iteration to speed up the convergence rate. By computing gradient and Hessian, the Newton's method holds a quadratic convergence rate. It requires to compute the second order differentials of a objective function and the computation is more complex at each iteration. Another drawback is that the Hessian calculation requires global information of all state elements. In this case, the distributed computation is difficult to be incorporated in the Newton's method. Practically, a tradeoff always exists in between first-order and second-order method. The gradient method requires less computational cost at each iteration. But more iterations are required to get to the converging point. Reversely, the Newton's method takes more efforts on the computation at each iteration while leading to less iterations. Basically, the gradient descent method is the default algorithm in practice. By evaluating the optimization performance in terms of computational time and the objective value, we could decide which method is more preferred.

CHAPTER 3. PROBLEM DECOMPOSITION AND DISTRIBUTED OPTIMIZATION ALGORITHM

3.1 Introduction

Centralized optimization algorithms rely on a single computational node and centralized information. This implies many drawbacks when dealing with large scale problems with large datasets. First, the computational complexity could be extremely high due to the large datasets and high dimension of state variables. Handling such huge datasets and high dimensional problems requires not only the usefulness, but also a high quality of the single processor. Second, in some scenarios, it is difficult to collect and store data in a centralized manner. Even it can be done so, in cases of multicast, this could be time-consuming and application-restricted due to the bandwidth limitation. Third, when a single point failure occurs in the single computational node, it will cause task suspension.

Instead of using centralized optimization algorithm in a single-agent system, we introduce a multi-agent system where a connected network is established by building communications/connections among agents. This system efficiently improves the computational performance such that it can be implemented in large-scale problems. First, multi-agent system greatly reduces the scale of each sub-problem. It dramatically decreases the computational complexity for solving sub-problem associated with each single agent. To handle a large-scale problem, the multi-agents system is more scalable than a centralized system. It is more beneficial to employ a bunch of low performance computing elements than a single high performance computing workstation in terms of computation efficiency and cost. Second, agents are distributed for local data collection and storage. Local data processing reduces

time delay due to the bandwidth limitation and long transferring distance. Third, fault tolerance can be improved, i.e. a single point malfunction will not fail the entire system. If some agents crash due to hardware, software, or communication malfunctions, the assigned task can still be accomplished using the remaining working agents. Based on multi-agent systems, distributed optimization algorithms are developed in a parallel/sequential computation manner. In this section, we review typical decomposition methods, dual decomposition, as well as the calculus-based first-order and second-order optimization algorithms embedded in multi-agents system.

Dual problem, in most cases referred to as Lagrangian dual problem, sometimes leads to an efficient or distributed method to solve the original problem[Boyd and Vandenberghe (2004)]. One famous implementation example is the Support Vector Machine (SVM). Solving associated dual problem successfully avoid constructing a proper mapping function which is extremely difficult in practice. Instead, SVM relies on kernel function with deterministic pattern and unknown parameters to be determined. Learning process is to solve a dual problem subject to a set of inequality constraints [Bishop (2006)]. Dual problem is generated by introducing Lagrangian relaxation. Solving dual problem provides the optimal solution for dual variables. Then it is converted into corresponding primal solution through an optimizer. Minimizing primal objective is equal to maximizing dual objective with fewer constraints (at least nonnegativity for dual variables) if strong duality holds. By constructing dual problem, a separable Lagrangian can be decomposed into a sequence of sub-Lagrangians to generate the dual sub-problems. This process is referred to as dual decomposition. Dual problem formulation and associated dual decomposition method are discussed in Section 3.2.

Calculus-based distributed methods can be applied to dual-decomposed sub-problems or directly to primal problems with a distributed computation scheme. In the rest of this section, we describe four typical distributed optimization methods, i.e. dual ascent/subgradient, projected subgradient, ADMM, and Newton-type distributed method. For each method,

we first provide the algorithm description and then analyze the convergence property by giving relative proofs.

Dual ascent, a.k.a. dual subgradient method, is one of the classical decentralized algorithms based on dual decomposition. It consists a x-minimization and a dual-updating step in each iteration. At x-minimization step, local primal variables are optimized by minimizing the decomposed Lagrangian independently. There are many ways to design a local optimizer. For example, as a first-order method, the steepest decent algorithm leads to a linear convergence rate, which only relies on the gradient information. However, the Newton's method with quadratic convergence rate, requires Hessian matrix calculation in addition to the gradient. In practice, a tradeoff needs to be considered between computational simplicity with slow convergence speed and complex computation with fast convergence speed. At the second step, we update the dual variables using current primal solutions. We repeat these two steps recursively until the primal or dual variables converges. Algorithm and convergence analysis are discussed in Section 3.3.

As an extension of subgradient method, projected subgradient method can be applied to either primal or dual problem. When directly applying to a primal problem, it projects current point to the convex feasible region by using an Euclidean projection operator, which is followed by updating step of the primal variables. There is no requirement of designing the optimizer. Primal iteration only depends on simple mathematical computation. In this case, it typically requires much more iterations than those algorithms based on optimizer design. When applied to the dual problem, the projected subgradient method projects dual variables to a non-negative region. We handle this non-negative projection as a dual variable updating step in the standard subgradient method in Section 3.3. Algorithm and convergence analysis are discussed in Section 3.4.

ADMM is an efficient and simple distributed algorithm for solving convex optimization problems in an sequential manner. It follows decomposition-coordination procedure and progressively solve each sub-problem with local information required. ADMM combines

the benefits from the method of multipliers and dual decomposition. First, by constructing augmented Lagrangian, it converts a non-strict-convex objective to a strict one by introducing additional quadratic relaxation terms in the original objective. Comparing to the Lagrangian, augmented Lagrangian guarantees the primal convergence even through objective is not strictly convex or taking value of $+\infty$ [Boyd et al. (2011)]. Second, distributed computation is achieved by applying x-minimization steps in an sequential fashion after dual decomposition. Algorithm and convergence analysis are discussed in Section 3.5.

Inspired by the rapid convergence of the Newton's method, we propose a Newton-type distributed optimization algorithm, aiming at increasing the rate of convergence. Distributed Newton method consists of primal iteration and Newton iteration. Instead of using gradient as the direction factor in steepest descent method, a Newton direction is applied in primal iteration. Newton direction is updated by a function of gradient, Hessian, and current optimal solution of a Newton variable $\mathbf{w}$. Optimal Newton variable is obtained by Newton iteration. The convergence speed has been proved to be faster than dual subgradient method based on dual decomposition. Algorithm and convergence analysis are discussed in Section 3.6.

3.2 Dual Problem and Dual Decomposition

For an optimization problem in (1.1) to minimize an objective function $f(\mathbf{x})$ under equality constraints $h_i(\mathbf{x}) = 0$, $i = 1, \ldots, p$, and inequality constraints $g_j(\mathbf{x}) \leq 0$, $j = 1, \ldots, m$, its Lagrangian function is formulated as

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j g_j(\mathbf{x}), \tag{2.1}$$

where μ_i, $i = 1, \ldots, p$ and λ_j, $j = 1, \ldots, m$ are Lagrangian multipliers. If the objective and constraint functions can be expressed in the summation form of

$$\begin{aligned} f(\mathbf{x}) &= \sum_{k=1}^{K} f^k(\mathbf{x}_k) \\ g_i(\mathbf{x}) &= \sum_{k=1}^{K} g_i^k(\mathbf{x}_k) \\ h_j(\mathbf{x}) &= \sum_{k=1}^{K} h_j^k(\mathbf{x}_k) \end{aligned}$$

by partition the state vector $\mathbf{x}$ into subvectors $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_K)$, the Lagrangian is reformulated as

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{k=1}^{K} (f^k(\mathbf{x}_k) + \sum_{i=1}^{p} \mu_i h_i^k(\mathbf{x}_k) + \sum_{j=1}^{m} \lambda_j g_j^k(\mathbf{x}_k)).$$

The above function can be decomposed into K subproblems according to the subvector $\mathbf{x}^k$, where $k = 1, \ldots, K$. For each subproblem, it can be solved by minimizing kth Lagrangian $L^k(\mathbf{x}^k, \boldsymbol{\lambda}, \boldsymbol{\mu})$,

$$\begin{aligned} L_D^k(\boldsymbol{\lambda}, \boldsymbol{\mu}) &= \min_{\mathbf{x}^k} L^k(\mathbf{x}_k, \boldsymbol{\lambda}, \boldsymbol{\mu}) \\ &= \min_{\mathbf{x}^k} (f^k(\mathbf{x}_k) + \sum_{i=1}^{p} \mu_i h_i^k(\mathbf{x}_k) + \sum_{j=1}^{m} \lambda_j g_j^k(\mathbf{x}_k)), \end{aligned} \tag{2.2}$$

where $L_D^k(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is dual function for a given pair of multipliers $(\boldsymbol{\lambda}, \boldsymbol{\mu})$, which is always concave. Thus the dual problem

$$\begin{gathered} \max_{(\boldsymbol{\lambda}, \boldsymbol{\mu})} L_D^k(\boldsymbol{\lambda}, \boldsymbol{\mu}) \\ s.t.\ \boldsymbol{\lambda} \succeq 0 \end{gathered} \tag{2.3}$$

is a convex optimization problem.

3.3 Dual Ascent/Sub-Gradient Method

3.3.1 Dual Ascent/Sub-Gradient Method

Sub-gradient method is an iterative procedure to gradually approach the optimization solution by finding the ascent direction for the dual problem. Detailed procedures regarding

step size selection and convergence proof can be found in Boyd et al. (2003). At each sequence q, assuming the multipliers $(\boldsymbol{\lambda}^q, \boldsymbol{\mu}^q)$ are given, the subgradient at this point is expressed as

$$g_s(\mathbf{x}^q) = \begin{pmatrix} g(\mathbf{x}^q) \\ h(\mathbf{x}^q) \end{pmatrix}. \tag{3.4}$$

Then each sub-problem can be solved in parallel by the following x-minimization and Lagrangian updating:

$$\mathbf{x}_k^{q+1} = \arg\min_{\mathbf{x}^k} L^k(\mathbf{x}_k, \boldsymbol{\lambda}^q, \boldsymbol{\mu}^q) \tag{3.5}$$

$$\boldsymbol{\lambda}^{q+1} = \max(0, \boldsymbol{\lambda}^q + \alpha_\lambda^q g(\mathbf{x}^q)) \tag{3.6}$$

$$\boldsymbol{\mu}^{q+1} = \boldsymbol{\mu}^q + \alpha_\mu^q h(\mathbf{x}^q), \tag{3.7}$$

where α_λ^q and α_μ^q are the step size that will control the convergence speed of the subgradient method. The maximum number of sequence q is generally defined to satisfy the stopping criteria of iteration.

3.3.2 Convergence Analysis

Our goal in this section is to exhibit linear convergence for the distributed optimization algorithm using dual subgradient method. The following proof is based on linear convergence conclusion described in Terelius (2010) for iterative subgradient method. The conclusion related to error upper bound is still valid for dual subgradient method. To verify the linear convergence of the primal varialbe x, we further introduce two assumptions and convergence results regarding the dual subgradient method.

Assumption 3.3.1. *The norm of dual subgradient $g_s(\mathbf{x})$ at each iteration q is bounded by G such that*

$$\| g_s(\mathbf{x}) \| \leq G, \quad \forall \mathbf{x} \in \mathbb{R}^n. \tag{3.8}$$

Assumption 3.3.2. *The distance, denoted as D, from the initial primal state $\mathbf{x}^0$ to the optimal set $\mathbf{x}^*$ is bounded by U such that*

$$D(\mathbf{x}^0, \mathbf{x}^*) \leq U. \tag{3.9}$$

Theorem 3.3.1. *The difference between the best result $J_{best}^{q_m}$ and objective value J^* at optimal points from dual subgradient method is bounded by*

$$J_{best}^{q_m} - J^* \leq \frac{U^2 + G^2 \sum_{q=0}^{q_m} (\alpha^q)^2}{2 \sum_{q=0}^{q_m} \alpha^q}, \tag{3.10}$$

where q_m is the maximum number of iteration [Terelius (2010)]. The detailed proof can be found in Appendix A.

Proposition 3.3.2. *Let r and β be constant and $0 < \beta < 1$. The dual subgradient method converges to the optimal solution x^* with linear rate bounded by $\frac{U^2(1-\beta)}{2r} + \frac{G^2 r}{2(1+\beta)}$ using step size $\alpha^q = r(\beta)^q$.*

Proof. Following the conclusion in 3.3.1 and substituting $\alpha^q = r(\beta)^q$ into Eq.(3.10), the difference between $J_{best}^{q_m}$ and J^* is bounded by

$$\begin{aligned} J_{best}^{q_m} - J^* &\leq \frac{U^2 + G^2 r^2 \frac{1-\beta^{2q_m}}{1-\beta^2}}{2r\frac{1-\beta^{q_m}}{1-\beta}} \\ &= \frac{U^2(1-\beta^2) + G^2 r^2 (1-\beta^{2q_m})}{2r(1-\beta^{q_m})(1+\beta)} \\ &= \mathcal{J}^{q_m} \end{aligned} \tag{3.11}$$

As $q_m \to \infty$, above convergence result is simplified as

$$\begin{aligned} \lim_{q_m \to \infty} J_{best}^{q_m} - J^* &\leq \lim_{q_m \to \infty} \mathcal{J}^{q_m} \\ &= \frac{U^2(1-\beta)}{2r} + \frac{G^2 r}{2(1+\beta)} \\ &= \mathcal{J}_{inf} \end{aligned} \tag{3.12}$$

which indicates that the subgradient method using step $\alpha^q = r(\beta)^q$ converges to a region within $\frac{U^2(1-\beta)}{2r} + \frac{G^2 r}{2(1+\beta)}$ of optimality. By the definition of convergence rate, one has

$$\begin{aligned} \lim_{q_m \to \infty} \frac{|\mathcal{J}^{q_m} - \mathcal{J}_{inf}|}{|\mathcal{J}^{q_m-1} - \mathcal{J}_{inf}|} &= \lim_{q_m \to \infty} \frac{U^2(1-\beta^2) + G^2 r^2 (1-\beta^{q_m})}{U^2(1-\beta^2)\beta^{-1} + G^2 r^2 (1-\beta^{q_m-1})\beta^{-1}} \cdot \frac{1-\beta^{q_m-1}}{1-\beta^{q_m}} \\ &= \beta. \end{aligned} \tag{3.13}$$

Since $\beta \in (0,1)$, sequence $\mathcal{J}^{q_m}$ converges linearly to $\mathcal{J}_{inf}$, i.e. sequence $J_{best}^{q_m}$ R-linearly converges to J^* within $\frac{U^2(1-\beta)}{2r} + \frac{G^2 r}{2(1+\beta)}$ of optimality. □

From Theorem 3.3.1 and Proposition 3.3.2, they indicate that the convergence speed and accuracy is highly related to the step size α^q. For a significantly small β, it introduces a large neighborhood around the optimal solution, which reduces the estimation accuracy. On the other hand, if β is approaching to one, the step size becomes a constant. In cases when a large value of r is used, it might lead to a divergent result. To balance between convergence and precision, we select a relatively large step size at the beginning and decreases it progressively along each iteration.

3.4 Projected Subgradient Method

3.4.1 Projected Subgradient Method

As an extension of subgradient method, projected subgradient method makes a projection after updating state value such that the current primal solution locates in the feasible region. To solve the following convex optimization problem constrained by a convex set C expressed as

$$\begin{aligned} &\min f(\mathbf{x}) \\ &s.t.\ \mathbf{x} \in C, \end{aligned} \tag{4.14}$$

the projected subgradient method maps state value on C by employing Euclidean projection P, expressed by

$$\mathbf{x}^{q+1} = P(\mathbf{x}^q - \alpha^q g_s(\mathbf{x}^q)), \tag{4.15}$$

where $g_s(\mathbf{x}^q)$ is the subgradient of f at $\mathbf{x}^q$. In some cases, projection P only relies on local information. For example, if the convex set $C = \{\mathbf{x}|A\mathbf{x} = b\}$ is linear, projection P is linear as well and can be expressed as [Boyd et al. (2003)]

$$P(\mathbf{y}) = \mathbf{y} - A^T(AA^T)^{-1}(A\mathbf{y} - b) \tag{4.16}$$

Replacing P with Eq.(4.16) in Eq.(4.15), the updating step is expressed by

$$\mathbf{x}^{q+1} = \mathbf{x}^q - \alpha^q(I - A^T(AA^T)^{-1}A)g_s(\mathbf{x}^q). \tag{4.17}$$

Matrix $(I - A^T(AA^T)^{-1}A)$ is sparse and non-zero entries appear at certain locations to represent associated local information of subgradient $g_s(\mathbf{x}^q)$. Therefore, primal iteration relies on local information of state vector and subgradient. More specifically, if the convex objective function follows the summation form of $f(\mathbf{x}) = \sum_{k=1}^{K} f_k(\mathbf{x}_k)$ and the feasible convex set is linear as follows,

$$\begin{aligned} &\min \sum_{k=1}^{K} f_k(\mathbf{x}_k) \\ &s.t.\ A\mathbf{x} = b. \end{aligned} \tag{4.18}$$

Global optimum can be obtained through projected subgradient method in a distributed scheme as follows,

---

Initialization: Solve each unconstrained subproblem independently and obtain the each optimal state with the initial value $\mathbf{x}_k^0$ *and* $q = 1$.

while $q \leq q_{max}$ *or not converged* **do**

  for $k \leftarrow 1$ **to** K *(in parallel)* **do**

    $\mathbf{x}_k^{q+1} = P_k(\mathbf{x}^q - \alpha^q g_s(\mathbf{x}^q))$

    $q = q + 1$

  end

end

Protocol 3 Solving 4.18 Using Projected Subgradient Method

In the above algorithm, P_k is the projection operator for subproblem k. The projected subgradient method can be implemented in solving dual sub-problem (2.3). In this case, the update step for multiplier $\boldsymbol{\lambda}$ follows Eq.(3.6) which projects $\boldsymbol{\lambda}$ to the feasible region, i.e. $\boldsymbol{\lambda} \succeq 0$.

3.4.2 Convergence Analysis

For the updating step before projection, i.e. $\mathbf{z}^{q+1} = \mathbf{x}^q - \alpha^q g_s(\mathbf{x}^q)$, we have [Boyd et al. (2003)]

$$\begin{aligned} ||\mathbf{z}^{q+1} - \mathbf{x}^*||_2^2 &= ||\mathbf{x}^q - \alpha^q g_s(\mathbf{x}^q) - \mathbf{x}^*||_2^2 \\ &= ||\mathbf{x}^q - \mathbf{x}^*||_2^2 - 2\alpha^q g_s(\mathbf{x}^q)(\mathbf{x}^q - \mathbf{x}^*) + (\alpha^q)^2 ||g_s(\mathbf{x}^q)||_2^2 \\ &\leq ||\mathbf{x}^q - \mathbf{x}^*||_2^2 - 2\alpha^q g_s(\mathbf{x}^q)(f(\mathbf{x}^q) - f^*) + (\alpha^q)^2 ||g_s(\mathbf{x}^q)||_2^2. \end{aligned} \tag{4.19}$$

By following the proof of Theorem 3.3.1, x converges to the optimal solution $\mathbf{x}^*$ without projection. After projection using operator P, we notice that

$$\begin{aligned} ||\mathbf{x}^{q+1} - \mathbf{x}^*||_2^2 &= ||P(\mathbf{z}^{q+1}) - \mathbf{x}^*||_2^2 \\ &\leq ||\mathbf{z}^{q+1} - \mathbf{x}^*||_2^2. \end{aligned} \tag{4.20}$$

The inequality holds due to the fact that a smaller Euclidean distance to the optimal value $\mathbf{x}^*$ is obtained after projecting $\mathbf{z}^{q+1}$ from an infeasible region to a feasible region. If $\mathbf{z}^{q+1}$ locates in a feasible region, the operator P has no effect on $\mathbf{z}^{q+1}$, i.e. $\mathbf{x}^{q+1} = \mathbf{z}^{q+1}$. Then the equality holds for the second line. We conclude that projection P reduces the Euclidean distance between next step state value $\mathbf{x}^{q+1}$ and the optimal one $\mathbf{x}^*$, which proves the convergence of the projected subgradient method. Linear convergence rate can be proved by following the proofs in Section 3.3.2.

3.5 Alternating Direction Method of Multipliers

3.5.1 Alternating Direction Method of Multipliers

ADMM combines the decomposability of dual subgradient method and the superior convergence of the method of multiplier [Boyd et al. (2011)]. It provides a fast convergence rate with modest accuracy. The problem in the following form can be solved by ADMM,

$$\begin{aligned} min. \quad & \sum_{k=1}^{k=K} f_k(\mathbf{x}_k) \\ s.t. \quad & \sum_{k=1}^{k=K} A_j^k \mathbf{x}_k = \mathbf{b}_j, \ j = 1, ..., J. \end{aligned} \tag{5.21}$$

The objective in (5.21) is expressed by the summation of K sub-objectives. State vector is partitioned into K subvectors where each of them is associated with a sub-objective, i.e. $f_k(\mathbf{x}_k)$. $\sum_{k=1}^{k=K} A_j^k \mathbf{x}_k = \mathbf{b}_j$ represents jth linear equality constraint and we have J of them in total. The augmented Lagrangian is formulated as

$$\begin{aligned} L_a(\mathbf{x}, \boldsymbol{\mu}) &= \sum_{k=1}^{k=K} f_k(\mathbf{x}_k) + \sum_{j=1}^{j=J} \boldsymbol{\mu}_j^T (\sum_{k=1}^{k=K} A_j^k \mathbf{x}_k - \mathbf{b}_j) \\ &\quad + \frac{\rho}{2} \sum_{j=1}^{j=J} || \sum_{k=1}^{k=K} A_j^k \mathbf{x}_k - \mathbf{b}_j ||_2^2 \\ &= \sum_{k=1}^{k=K} [f_k(\mathbf{x}_k) + \sum_{j=1}^{j=J} (\boldsymbol{\mu}_j^T A_j^k \mathbf{x}_k + \frac{\rho}{2} h_j^k(\mathbf{x}_k))] - \sum_{j=1}^{j=J} \boldsymbol{\mu}_j^T \mathbf{b}_j \\ &= \sum_{k=1}^{k=K} L_k(\mathbf{x}_k, \mathbf{x}_S, \boldsymbol{\mu}) - \sum_{j=1}^{j=J} \boldsymbol{\mu}_j^T \mathbf{b}_j \end{aligned} \tag{5.22}$$

where $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^T, ..., \boldsymbol{\mu}_J^T]^T$ is the Lagrangian multiplier vector.

We let $\sum_{k=1}^{k=K} h_j^k(\mathbf{x}_k) = || \sum_{k=1}^{k=K} A_j^k \mathbf{x}_k - b_j ||_2^2$ and assume $\mathbf{x}_S$ is a subset of the collection of the state vectors excluding $\mathbf{x}_k$. $\mathbf{x}_S$ contains the local information associated with x_k due to the zero matrix for $A_j^{k'}$, where $k' \notin \{k, S\}$. $\mathbf{x}_S$ and $\boldsymbol{\mu}$ are numerical vector values in the following iteration steps. For each subproblem, the x-minimization step updates x_k as follows

$$\arg\min_{\mathbf{x}_k} L_k(\mathbf{x}_k, \mathbf{x}_S, \boldsymbol{\mu}). \tag{5.23}$$

ADMM offers a effective way to approach to the optimal solutions in an iterative manner. ADMM is similar to dual subgradient method excluding the additional quadratic term in augmented Lagrangian (5.22). It consists of a sequential state updating at each iteration. As the step size in dual subgradient method, we employ augmented Lagragian parameter ρ to update dual variable $\boldsymbol{\mu}$ and control the convergence speed and accuracy. ADMM for solving problem (5.21) can be summerized as

---

Initialization: Solve each subproblem independently with $\boldsymbol{\mu} = \bar{0}$ *and obtain the each optimal state as the initial value* $\mathbf{x}_k^0$.

while $p \leq p_{max}$ **do**

  for $k \leftarrow 1$ **to** K **do**

    $\mathbf{x}_k^{q+1} = \arg\min_{\mathbf{x}_k} L_k(\mathbf{x}_k, \mathbf{x}_S^q, \boldsymbol{\mu}^q)$.

    Update $\mathbf{x}_k^q = \mathbf{x}_k^{q+1}$.

  end

  Update $\boldsymbol{\mu}$ locally: $\boldsymbol{\mu}_j^{q+1} = \boldsymbol{\mu}_j^q + \rho(\sum_{k=1}^{k=K} A_j^k \mathbf{x}_k^{q+1} - \mathbf{b}_j)$.

  $q = q + 1$

end

Protocol 4 Solving 5.21 Using ADMM Iterations

3.5.2 Convergence Analysis

To verify the algorithm convergence, two assumptions are introduced. For problem (5.21), we have the following assumptions regarding each sub-problem and unaugmented Lagrangian [Boyd et al. (2011)].

Assumption 3.5.1. *The objective functions* $f_k(\mathbf{x}_k)$, $k = 1, ..., K$, *are closed, proper, and convex.*

Assumption 3.5.1 implies that each sub-problem is solvable. The augmented Lagrangian can be minimized by x-iteration step sequentially.

Assumption 3.5.2. *The unaugmented Lagrangian L has a saddle point, i.e.* $L(\mathbf{x}_1^*, ..., \mathbf{x}_K^*, \boldsymbol{\mu}) \leq L(\mathbf{x}_1^*, ..., \mathbf{x}_K^*, \boldsymbol{\mu}^*) \leq L(\mathbf{x}_1, ..., \mathbf{x}_K, \boldsymbol{\mu}^*)$ *holds* $\forall \mathbf{x}_k$, $k = 1, ...K$ *and* $\boldsymbol{\mu}$.

Assumption 3.5.2 indicates that the value of L on saddle point $(\mathbf{x}_1^*, ..., \mathbf{x}_K^*, \boldsymbol{\mu}^*)$ is finite. We can conclude that $(\mathbf{x}_1^*, ..., \mathbf{x}_K^*)$ satisfies equality constraints in (5.21) and it is a solution to (5.21). Moreover, the strong duality holds, which means the solution $(\mathbf{x}^{1*}, ..., \mathbf{x}^{K*})$ to dual problem (5.22) are equal to primal problem (5.21). Define the primal residual as $\mathbf{r}_j^q = \sum_{k=1}^K A_j^k \mathbf{x}_k^q - \mathbf{b}_j$ for each linear constraint. The convergence of primal residual has been proved by Stephen Boyd in Boyd et al. (2011), i.e. $\mathbf{r}_j^q \to 0$ as $q \to \infty$. Based on Stephen Boyd's paper, we extend the case to solving K subproblems and prove the objective convergence in Appendix B.

3.6 Distributed Newton's Method

Inspired by the rapid convergence of the Newton's method in solving network utility maximization problems [Dolev et al. (2009); Wei et al. (2010)], we propose a Newton-type distributed optimization algorithm, aiming at increasing the rate of convergence.

3.6.1 Distributed Newton Method

For a network consensus problem defined as

$$\begin{aligned} min. \quad & f(\mathbf{x}) \\ s.t. \quad & \mathbf{x}_i - \mathbf{x}_j = 0,\ i, j = 1, ..., K. \end{aligned} \tag{6.24}$$

where $\mathbf{x_k} \in \mathbb{R}^n$ and $\mathbf{x} \in \mathbb{R}^{n'}$, $n' = Kn$. From a feasible starting point $\mathbf{x}^0$, the iterative Newton approach for solving the constrained optimization problem is expressed as

$$\mathbf{x}^{q+1} = \mathbf{x}^q + s^q \Delta \mathbf{x}^q, \tag{6.25}$$

where $\Delta \mathbf{x}^q$ and s^q are the Newton direction and step size, respectively, at iteration step q.

We incorporate consensus constraints to compact form of $C\mathbf{x} = 0$. The Newton direction $\Delta\mathbf{x}^q$ is obtained by solving the following linear function,

$$\begin{pmatrix} \nabla^2 f(\mathbf{x}^q) & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x}^q \\ \mathbf{w}^q \end{pmatrix} = - \begin{pmatrix} \nabla f(\mathbf{x}^q) \\ 0 \end{pmatrix}, \tag{6.26}$$

where $\nabla^2 f(\mathbf{x}^q)$ and $\nabla f(\mathbf{x}^q)$ are the Hessian matrix and the gradient of the objective function evaluated at $\mathbf{x}^q$, respectively, and $\mathbf{w}^q$ is the dual variable of the linear constraint. For notation simplicity, we use $\nabla^2 f^q = \nabla^2 f(\mathbf{x}^q)$ and $\nabla f^q = \nabla f(\mathbf{x}^q)$ in the following text. From (6.26), we get

$$\Delta\mathbf{x}^q = -(\nabla^2 f^q)^{-1}((\nabla f^q) + C^T\mathbf{w}^q) \tag{6.27}$$

$$(C(\nabla^2 f^q)^{-1}C^T)\mathbf{w}^q = -C(\nabla^2 f^q)^{-1}\nabla f^q. \tag{6.28}$$

$\mathbf{w}^q$ can be solved by (6.28) through the decentralized method in Wei et al. (2010). Therefore, we can find the estimates in a distributed manner. The major steps of obtaining $\mathbf{w}^q$ is described below and more details can be referred to Wei et al. (2010). We first split the matrix $C(\nabla^2 f^q)^{-1}C^T$ as

$$C(\nabla^2 f^q)^{-1}C^T = (D^q + \bar{B}^q) + (B^q - \bar{B}^q), \tag{6.29}$$

where D^q is a diagonal matrix defined by $D^q = diag(C(\nabla^2 f^q)^{-1}C^T)$, $B^q = C(\nabla^2 f^q)^{-1}C^T - D^q$, and $\bar{B}^q$ is a diagonal matrix as well with diagonal entries defined as $(\bar{B}^q)_{bb} = \Sigma_{d=1}^{m'} B^q_{bd}$, $b = 1, \ldots, m'$.

In Eq.(6.28), $\mathbf{w}^q \in \mathbb{R}^{m'}$ consists of dual variable element corresponding to each proper link l connecting a pair of local agents for information exchanging. Let $w_l(0)$ be an arbitrary initial value of dual variables at each link l and the sequence $w_l(t)$ is generated by the following iterative manner

$$\begin{aligned} w_l(t+1) \quad = \quad & \frac{1}{(D^q)_{ll} + (\bar{B}^q)_{ll}}((\hat{B}^q)_{ll}w_l(t) - \Sigma_{i\in S(l)}\Pi_i(t) \\ & +\Sigma_{i\in S(l)}(\nabla^2 f^q)^{-1}_{ii}w_l(t) - \Sigma_{i\in S(l)}[(\nabla^2 f^q)^{-1}]_{ii}\nabla f^q_{x(i)}), \end{aligned} \tag{6.30}$$

where $\Pi_i(t) = (\nabla^2 f^q)^{-1}_{ii}\Sigma_{l\in\mathcal{E}(i)}w_l(t)$, $l \in \mathcal{E}(i)$ denotes to one of the links connecting agent i with the other agent, and agent i, $i \in S(l)$ is one of the agents connected by link l.

3.6.2 Convergence Analysis

The following discussion focuses on the convergence rate of the distributed estimation approach using Newton method. The goal is to prove the quadratic convergence rate of Newton method when applied in the constrained optimization problem. To obtain the quadratic convergence of Newton method, required conditions are stated below.

Theorem 3.6.1. *Consider exact Newton method characterized by Eq.(6.25). If the objective function $f(\mathbf{x})$ satisfies the following assumptions [Boyd and Vandenberghe (488)]:*
1. *The Hessian $H_e(x) = \nabla^2 f$ is Lipschitz continuous on $\mathbb{R}^{n'}$ with constant $L_c > 0$, i.e.*

$$\| H_e(\mathbf{x}) - H_e(\mathbf{y}) \|_2 \leq L_c \| \mathbf{x} - \mathbf{y} \|_2 \; \forall \; \mathbf{x}, \mathbf{y} \in \mathbb{R}^{n'}, \tag{6.31}$$

2. *The function $f(\mathbf{x})$ is strictly convex with constant M, i.e.*

$$\| H_e(\mathbf{x})^{-1} \|_2 \leq M, \tag{6.32}$$

3. *At time k, the norm of gradient is bounded by η, i.e. $\| g_k^j \|_2 \leq \eta$ where $0 \leq \eta \leq \frac{1}{M^2 L_c}$. Then the Newton method has quadratic convergence rate, denoted as*

$$f(\mathbf{x}_k^j) - f^* \leq \frac{2}{M^3 L_c^2} 2^{-2^j}. \tag{6.33}$$

To proceed with the demonstration of quadratic convergence, it is necessary to verify the existence of the above assumptions. The verification for the Lipschitz continuous Hessian of $f(\mathbf{x})$ is obvious since H_e expressed in objective of (6.24) is independent of $\mathbf{x}_k^q$. Once the 1-prediction primal state $\mathbf{x}_k^-$ is obtained, Hessian can be derived before implementing the Newton method and remains constant during the entire Newton iteration. Therefore, $\exists \, L_c > 0$ such that

$$\| H_e(\mathbf{x}) - H_e(\mathbf{y}) \|_2 \equiv 0 \leq L_c \| \mathbf{x} - \mathbf{y} \|_2, \tag{6.34}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n'}$ and the equality holds when $\mathbf{x} = \mathbf{y}$. For the same reason, the 2-norm of H_e^{-1} has an upper bound M which can be obtained from the largest eigenvalue of matrix

$(H_e^T H_e)^{-1}$, denoted as $M \geq \sqrt{\lambda_{max}(H_e^T H_e)^{-1}}$. Based on the fact that inequality holds in Eq.(6.34) even if L_c is sufficiently close to zero, L_c can be selected as a sufficiently positive small value. Furthermore, to achieve fast convergence rate in Eq.(6.33), the smallest value of M can be set as $M = \sqrt{\lambda_{max}(H_e^T H_e)^{-1}}$. Thus $\frac{1}{M^2 L_c}$ can be sufficiently large such that $\exists\, \eta$ satisfying $0 < \eta \leq \frac{1}{M^2 L_c}$. This conclusion leads to $\| g_k^j \|_2 < \eta$. Hence, we have theoretically verified that the Newton method makes the convergence of primal solution faster than the dual subgradient method when applying to the distributed estimation problem.

3.7 Conclusion

In this section, we introduce calculus-based distributed method for solving convex optimization problem. By Lagrangian relaxation, original constrained problems are reformulated to be unconstrained problems. If both objective and constraint functions follow the summation form, original problem can be divided into a set of subproblems by dual decomposition. Four typical distributed methods are discussed. First, dual ascent/sub-gradient method relies on gradient/sub-gradient computation. The primal solution converges to optimum with a linear convergence rate. Second, projected subgradient method can be implemented directly in primal problem or dual problem, which leads to distributed computation scheme in some cases. Third, ADMM combines the decomposability of dual sub-gradient method and the superior convergence of method of multiplier. It is a simple but effective method for problems whose objective is not strictly convex or taking value of $+\infty$, e.g. a linear objective. At last, Newton-type method relies on gradient and Hessian calculation. It improves the convergence to a quadratic rate with additional Hessian requirement. However, this requires a twice differentiable objective function. In practice, there always be a tradeoff between fast convergence (quadratic rate) with more information (state values, gradient and Hessian) , and slow convergence (linear rate) with less information (state values and gradient).

CHAPTER 4. DISTRIBUTED OPTIMIZER DESIGN: DISTRIBUTED STATE ESTIMATION BASED ON EXTENDED KALMAN FILTER

4.1 Problem Statement

The rotational and translational motion, including 3D position (x, y, z), attitude $(pitch, roll, yaw)$, linear velocity (v_x, v_y, v_z) and angular velocity $(\omega_x, \omega_y, \omega_z)$, can be estimated through EKF in an single agent system. The observations are a sequence of 2D coordination on image plane provided by single sensor (camera). To improve the fault tolerance and estimation accuracy, we are required to implement a multi-sensor and multi-agent system. It turns out a sensor network where each vertex represents a combination of sensor and agent. The sensor-agent nodes can locally communicate with each other through wireless connection. The estimation results obtained from every agents are able to be consensus under the condition of different sensors with different measurement accuracy. Solving for such consensus optimization problem follows a distributed fashion and a recursive way.

4.2 Dual Quaternion, Kinematics and Dynamics

Dual quaternion is known due to the characteristics of compactness, geometrically meaningful representation, and non-singularity. It also greatly improves the computational efficiency in solving pose and position related estimation problems. Compared with alternative representations for spatial transformations, dual quaternions based operations significantly reduce computational complexity. For example, Euler angles based representations generates trigonometric entries and highly nonlinear terms in state transition and observation

matrices when using EKF. It has been verified that unit dual quaternion has 33% reduction on multiply operations and 26% reduction on plus operations for composition of two spatial transformation while using only 67% of storage resources compared to the matrix transform [Funda and Paul (1990); Funda et al. (1990)], not to mention the extra cost of computing trigonometric functions for elements in the rotational matrix of Euler angle representation. Hence we adopt dual quaternions to represent a spatial rigid motion with six degrees of freedom.

4.2.1 Quaternion

The classical quaternion definition is

$$\mathbf{q} = (q_0, \vec{q}), \tag{2.1}$$

where $\vec{q} \in \mathbb{R}^3$ and $q_0 \in \mathbb{R}$ are the vector part and scalar part of the quaternion, respectively. In the following, we use notation $\mathbb{H}$ to represent the set of four-dimensional vector, i.e. $\mathbf{q} \in \mathbb{H}$. A unit quaternion $\mathbf{q_u}$ with 2-norm equivalent to one can be used to represent a rotation of angle θ about a unit axis $\vec{n}$ in the form of

$$\mathbf{q_r} = (\cos\frac{\theta}{2}, \vec{n}\sin\frac{\theta}{2}). \tag{2.2}$$

4.2.2 Dual Quaternion

Dual quaternions, introduced by Clifford [Clifford (1873)], can present six-degrees-of-freedom rigid transformations by unifying translation and rotation into a single-state frame. Mathematically, a dual quaternion is defined as

$$\hat{\mathbf{q}} = \mathbf{q_r} + \mathbf{q_d}\varepsilon, \tag{2.3}$$

where $\mathbf{q_r} \in \mathbb{H}$ denotes the real part, $\mathbf{q_d} \in \mathbb{H}$ denotes the dual part, and ε is the dual unit with $\varepsilon^2 = 0$ but $\varepsilon \neq 0$.

A spatial transformation including both translations and rotations can be expressed as follows,

$$\mathbf{q_r} = \left(\cos\tfrac{\theta}{2}, \vec{n}\sin\tfrac{\theta}{2}\right) \tag{2.4}$$
$$\mathbf{q_d} = \tfrac{1}{2}\mathbf{q_r} \otimes \mathbf{t_b},$$

where $\mathbf{t_b} = (0, \vec{t}_b) \in \mathbb{H}$ is a quaternion composed of the position vector $\vec{t}_b \in \mathbb{R}^3$ expressed in the body frame and a zero scalar part and '$\otimes$' represents the quaternion multiplication.

4.2.3 Dual Quaternion Kinematics

We can find the kinematics of a spatial transformation in terms of dual quaternions, given as [Goddard and Abidi (1998)]

$$\dot{\mathbf{q}}_\mathbf{r} = \frac{1}{2}\mathbf{q_r} \otimes \boldsymbol{\omega_b}, \tag{2.5}$$

where $\boldsymbol{\omega_b} = (0, \vec{\omega}_b) \in \mathbb{H}$ and $\vec{\omega}_b = [\omega_x, \omega_y, \omega_z]^T \in \mathbb{R}^3$ is angular velocity of the rotating body evaluated in the body frame. Based on the above derivation of $\dot{\mathbf{q}}_\mathbf{r}$ and expression of $\mathbf{q_d}$ in Eq. (2.4), we then find the derivative of the dual part $\mathbf{q_d}$, expressed as

$$\begin{aligned}
\dot{\mathbf{q}}_\mathbf{d} &= \frac{1}{2}\dot{\mathbf{q}}_\mathbf{r} \otimes \mathbf{t_b} + \frac{1}{2}\mathbf{q_r} \otimes \dot{\mathbf{t}}_\mathbf{b} \\
&= \frac{1}{4}\mathbf{q_r} \otimes \boldsymbol{\omega_b} \otimes \mathbf{t_b} + \frac{1}{2}\mathbf{q_r} \otimes \dot{\mathbf{t}}_\mathbf{b} \\
&= \frac{1}{2}\mathbf{q_r} \otimes \left(\frac{1}{2}\boldsymbol{\omega_b} \otimes \mathbf{t_b}\right) + \frac{1}{2}\mathbf{q_r} \otimes \dot{\mathbf{t}}_\mathbf{b} \\
&= \frac{1}{2}\mathbf{q_r} \otimes \left(\begin{bmatrix} 0 \\ \vec{\omega}_b \times \vec{t}_b \end{bmatrix} + \frac{1}{2}\mathbf{t_b} \otimes \boldsymbol{\omega_b}\right) + \frac{1}{2}\mathbf{q_r} \otimes \dot{\mathbf{t}}_\mathbf{b} \\
&= \frac{1}{2}\mathbf{q_r} \otimes \left(\begin{bmatrix} 0 \\ \dot{\vec{t}}_b + \vec{\omega}_b \times \vec{t}_b \end{bmatrix} + \frac{1}{2}\mathbf{t_b} \otimes \boldsymbol{\omega_b}\right) \\
&= \frac{1}{2}\mathbf{q_r} \otimes \mathbf{v_b} + \frac{1}{4}\mathbf{q_r} \otimes \mathbf{t_b} \otimes \boldsymbol{\omega_b}
\end{aligned} \tag{2.6}$$

In above expressions, $\mathbf{v_b} = \begin{bmatrix} 0 \\ \dot{\vec{t}}_b + \vec{\omega}_b \times \vec{t}_b \end{bmatrix} = (0, \vec{v}_b) \in \mathbb{H}$ is a quaternion composed of the velocity vector $\vec{v}_b = [v_x, v_y, v_z]^T \in \mathbb{R}^3$ in the body frame and a zero scalar part. The forth equality in (2.6) is obtained via the definition of quaternion multiplication,

$$\begin{aligned} \frac{1}{2}\boldsymbol{\omega_b} \otimes \mathbf{t_b} &= \begin{bmatrix} -\frac{1}{2}\vec{\omega}_b \cdot \vec{t}_b \\ \frac{1}{2}\vec{\omega}_b \times \vec{t}_b \end{bmatrix} \\ &= \begin{bmatrix} -\frac{1}{2}\vec{\omega}_b \cdot \vec{t}_b \\ \vec{\omega}_b \times \vec{t}_b + \frac{1}{2}\vec{t}_b \times \vec{\omega}_b \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \vec{\omega}_b \times \vec{t}_b \end{bmatrix} + \begin{bmatrix} -\frac{1}{2}\vec{\omega}_b \cdot \vec{t}_b \\ \frac{1}{2}\vec{t}_b \times \vec{\omega}_b \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \vec{\omega}_b \times \vec{t}_b \end{bmatrix} + \frac{1}{2}\mathbf{t_b} \otimes \boldsymbol{\omega_b}. \end{aligned} \tag{2.7}$$

By replacing term $\frac{1}{2}\boldsymbol{\omega_b} \otimes \mathbf{t_b}$ in the third equality of (2.6) with the result of (2.7), it leads to the forth equality in (2.6).

4.2.4 Rigid Body Dynamics

The translational and rotational motion of a fully actuated rigid body can be described by the change rate of linear and angular momentum [Stengel (164)] in the form of

$$\vec{F} = \left[\frac{d}{dt}(m\vec{v})\right]_B = m \cdot \vec{v}_b + \vec{\omega}_b \times m\vec{v}_b \tag{2.8}$$

$$\vec{T} = \left[\frac{d}{dt}(J\vec{\omega})\right]_B = J\dot{\vec{\omega}}_b + \vec{\omega}_b \times J\vec{\omega}_b, \tag{2.9}$$

where $[d(\cdot)/dt]_B$ denotes the time derivative in the body frame, $m \in \mathbb{R}$ is the mass of the rigid body, $J \in \mathbb{R}^{3\times3}$ is the inertia tensor. F and T represent the net force and torque, respectively. As we all known that for a general three-dimensional body, it is always possible to find three mutually orthogonal axis, i.e. the principal axis for which the products of inertia are zero. Without losing generality, inertia tensor J therefore can be represented

as a diagonal matrix with the moment of inertia as $J = \text{diag}\,[\,J_{xx},\, J_{yy},\, J_{zz}\,]$. To further simplify the estimation process, the three principal axis fixed with rigid body is considered as the body frame.

4.3 Position Tracking and State Estimation Based on EKF in Single-Agent System

4.3.1 Introduction to Extended Kalman Filter

Consider a continuous nonlinear system with system dynamics $\dot{\mathbf{x}} = f(\mathbf{x}) + \mathbf{w}$ and an observation function $\mathbf{z} = h(\mathbf{x}) + \mathbf{v}$, where $\mathbf{x} \in \mathbb{R}^n$ are the states, $\mathbf{z} \in \mathbb{R}^m$ are the measurements, $\mathbf{w} \in \mathbb{R}^n$ is system noise with zero mean Gaussian sequence and covariance $Q \in \mathbb{R}^{n\times n}$, and $\mathbf{v} \in \mathbb{R}^m$ is measurement noise with zero mean Gaussian sequence and covariance $R \in \mathbb{R}^{m\times m}$. By discretizing and linearization, we can find the corresponding discrete linear system

$$\mathbf{x}^{t+1} = F^t\mathbf{x}^t + \mathbf{w}^t, \tag{3.10}$$

where $F^t = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\,|_{\mathbf{x}=\hat{\mathbf{x}}^t} \in \mathbb{R}^{n\times n}$ which is a state transition matrix at time interval t can be calculated by taking the partial derivative about the estimate $\hat{\mathbf{x}}^t$. The discrete observation model is

$$\mathbf{z}^t = H_k\mathbf{x}^t + \mathbf{v}^t, \tag{3.11}$$

where $H^t = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\,|_{\mathbf{x}=\mathbf{x}^{t-}} \in \mathbb{R}^{m\times n}$ is an observation matrix obtained by taking partial derivative about 1-step prediction $\mathbf{x}^{t-}$ prior to the input of observations. Different from standard Kalman filer which has constant state and observation matrices, the state and observation matrices in EKF are updated at each time interval t.

The EKF algorithm for the system described by Eqs.(3.10) and (3.11) finds the maximum-likelihood estimations $\hat{x}_k$ by minimizing the following least-square objective function [Bryson (1975)],

$$f(\mathbf{x}^t) = \frac{1}{2}[(\mathbf{x}^t - \mathbf{x}^{t-})^T(P^{t-})^{-1}(\mathbf{x}^t - \mathbf{x}^{t-}) + (\mathbf{z}^t - h(\mathbf{x}^t))^T R^{-1}(\mathbf{z}^t - h(\mathbf{x}^t)], \tag{3.12}$$

where $P^{t-} = E[(\mathbf{x}^t - \mathbf{x}^{t-})(\mathbf{x}^t - \mathbf{x}^{t-})^T]$ is the error covariance prior to incorporating the measurement at time t into the estimation. Given all the information up to time t in terms of the prediction state $\mathbf{x}^{t-}$ and the measurement $\mathbf{z}^t$, the optimal solution of Eq. (3.12) provides a recursive estimation $\hat{\mathbf{x}}^t$ as following,

$$\hat{\mathbf{x}}^t = \mathbf{x}^{t-} + K(\mathbf{z}^t - h(\mathbf{x}^{t-})), \tag{3.13}$$

with Kalman gain expressed as $K = P^t H^t R^{-1}$. P^t in the Kalman gain expression is the covariance matrix of the error vector $\mathbf{x}^t - \hat{\mathbf{x}}^t$, thus

$$\begin{aligned} P^t &= E[(\mathbf{x} - \hat{\mathbf{x}}^t)(\mathbf{x} - \hat{\mathbf{x}}^t)^T] \\ &= ((P^{t-})^{-1} + H^{tT} R^{-1} H^t)^{-1}, \end{aligned} \tag{3.14}$$

where $P^{t-} = F^{t-1} P^{t-1} F^{t-1T} + Q$.

The key elements before applying EKF are to construct the state transition and observation models. State transition model derived from dual kinematics indicates the evolution from current to future states. Observation model using single image sensor can be constructed through the relationship between two relative frames, the object frame and the camera frame. Before introducing these models, we first select the following states to represent the spatial rigid motion of a concerned object. They are

$$\mathbf{x} = [\mathbf{q_r}^T \, \mathbf{q_d}^T \, \boldsymbol{\omega_b}^T \, \mathbf{v_b}^T]^T. \tag{3.15}$$

4.3.2 State Transition Model Based on Dual Kinematics

By discretizing the dual kinematics expressed in Eqs. (2.5) - (2.9) with sampling time τ using forward Euler method, the state transition model associated with each element of $\mathbf{x}$ becomes [Olsson et al. (2003)]

$$\begin{aligned}
\mathbf{q_r}^{t+1} &= \mathbf{q_r}^t + \tfrac{\tau}{2}\mathbf{q_r}^t \otimes \boldsymbol{\omega_b}^t \\
\mathbf{q_d}^{t+1} &= \mathbf{q_d}^t + \tfrac{\tau}{2}\mathbf{q_r}^t \otimes \mathbf{v_b}^t + \tfrac{\tau}{4}\mathbf{q_r}^t \otimes \mathbf{t_b}^t \otimes \boldsymbol{\omega_b}^t. \\
\boldsymbol{\omega_b}^{t+1} &= \boldsymbol{\omega_b}^t + \tau(-\Omega_\omega \boldsymbol{\omega_b}^t + \begin{bmatrix} 0 \\ J^{-1}\vec{T}^t \end{bmatrix}) \\
\mathbf{v_b}^{t+1} &= \mathbf{v_b}^t + \tau(-\Omega_v \mathbf{v_b}^t + \tfrac{1}{m}\begin{bmatrix} 0 \\ \vec{F}^t \end{bmatrix})
\end{aligned} \tag{3.16}$$

By expanding the quaternion multiplication, we find the following relationship,

$$\begin{aligned}
\mathbf{q_r} \otimes \boldsymbol{\omega_b} &= S\boldsymbol{\omega_b} \\
\mathbf{q_r} \otimes \mathbf{v_b} &= S\mathbf{v_b}, \\
\mathbf{q_r} \otimes \mathbf{t_b} \otimes \boldsymbol{\omega_b} &= SM\boldsymbol{\omega_b}
\end{aligned} \tag{3.17}$$

where $S = \begin{bmatrix} q_0 & -\vec{q}^T \\ \vec{q} & q_0 I + K(\vec{q}) \end{bmatrix}$, $M = \begin{bmatrix} 0 & -\vec{t}_b^T \\ \vec{t}_b & K(\vec{t}_b) \end{bmatrix}$, the skew-symmetric matrix is defined as $K(\vec{y}) = \begin{bmatrix} 0 & -y_3 & y_2 \\ y_3 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{bmatrix}$. Furthermore, $\Omega_v = \begin{bmatrix} 0 & 0_{1\times 3} \\ 0_{3\times 1} & K(\omega_b) \end{bmatrix}$ and

$$\Omega_\omega = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{J_{zz}-J_{yy}}{J_{xx}}\omega_z & 0 \\ 0 & 0 & 1 & \frac{J_{xx}-J_{zz}}{J_{xx}}\omega_x \\ 0 & \frac{J_{yy}-J_{xx}}{J_{zz}}\omega_y & 0 & 1 \end{bmatrix}.$$

Assuming the net force and torque are known, based on the relationship developed in above, the state transition model can be written as compact form as follows.

$$\mathbf{x}^{t+1} = \begin{bmatrix} I_{4\times4} & 0_{4\times4} & \frac{\tau}{2}S & 0_{4\times4} \\ 0_{4\times4} & I_{4\times4} & \frac{\tau}{4}SM & \frac{\tau}{2}S \\ 0_{4\times4} & 0_{4\times4} & I_{4\times4} - \tau\Omega_\omega & 0_{4\times4} \\ 0_{4\times4} & 0_{4\times4} & 0_{4\times4} & I_{4\times4} - \tau\Omega_v \end{bmatrix}_k \mathbf{x}^t + \begin{bmatrix} 0_{9\times1} \\ (J^{-1}\vec{T})_{3\times1} \\ 0 \\ \frac{1}{m}\vec{F}_{3\times1} \end{bmatrix}. \tag{3.18}$$

4.3.3 Observation Model

The measurement instruments used here to estimate the spatial rigid motion of concerned object are image sensors/cameras. A set of feature points of the object will be identified before observing process. By recording the images of the feature points in the camera frame, we aim at estimating both translational and rotational motion of concerned object. Before we setup the observation model, we first find out the relationship between the object frame and the camera frame. Figure 4.1 demonstrates the coordinates of one feature point in body frame (x^o, y^o, z^o) and camera frame (x^c, y^c, z^c). The measurements of the object are obtained by observing its projection on the image plane, denoted as (x^i, y^i).

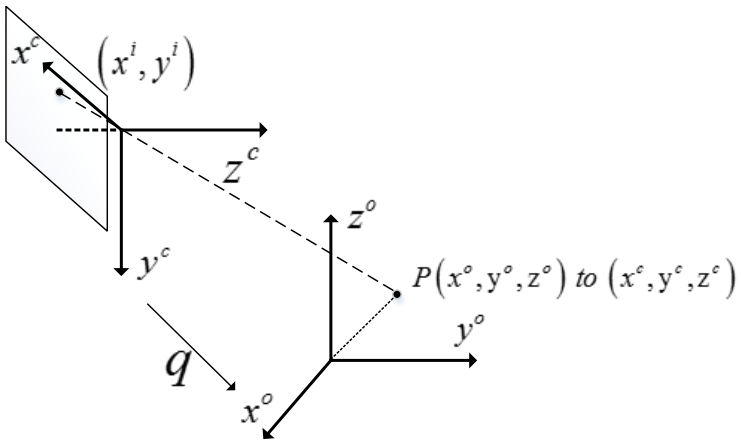

Figure 4.1 Observation of feature point in two reference frames.

By dual multiplication, a dual quaternion transformation between two frames is represented by

$$\hat{\mathbf{p}}^{\mathbf{c}} = \hat{\mathbf{q}} \hat{\otimes} \hat{\mathbf{p}}^{\mathbf{o}} \hat{\otimes} \hat{\mathbf{q}}^{*}, \tag{3.19}$$

where $\hat{\mathbf{p}}^{\mathbf{c}} = \mathbf{q_I} + (x^c i + y^c j + z^c k)\varepsilon$ and $\hat{\mathbf{p}}^{\mathbf{o}} = \mathbf{q_I} + (x^o i + y^o j + z_o k)\varepsilon$ denote the dual representation of the feature point in camera frame and object frame, respectively, with $\mathbf{q_I} = (1, 0i, 0j, 0k) \in \mathbb{H}$. $\hat{\mathbf{q}}^*$ represents the conjugate of dual quaternion $\hat{\mathbf{q}}$. Notation '$\hat{\otimes}$' refers to dual quaternion multiplication. The transformation process can be decomposed into two parts, pure translation and pure rotation. Under this assumption, Eq. (3.19) can be rewritten as

$$\hat{\mathbf{p}}^{\mathbf{c}} = \mathbf{q_I} + (\mathbf{q_r} \otimes \mathbf{p_d^o} \otimes \mathbf{q_r^*} + \mathbf{t_c})\varepsilon, \tag{3.20}$$

where $\mathbf{t_c}$ denotes translational motion of target in camera frame and $\mathbf{p_d^o} = (0, x^o i, y^o j, z^o k) \in \mathbb{H}$ represents the position quaternion in object frame. From the above expression, we can find the dual part of $\hat{\mathbf{p}}^{\mathbf{c}}$, which indicates the position of the feature point in camera frame, and rewrite it as

$$\mathbf{p_d^c} = \mathbf{q_r} \otimes \mathbf{p_d^o} \otimes \mathbf{q_r^*} + 2\mathbf{q_d} \otimes \mathbf{q_r^*}, \tag{3.21}$$

where $\mathbf{p_d^c} = (0, x^c i, y^c j, z^c k) \in \mathbb{H}$ represents the position quaternion in camera frame. The first term in the right side of Eq. (3.21) describes the rotation of the feature point and the second term indicates its translation. Corresponding projection of the feature point on 2-D image plane are obtained by [Wang and Wilson (1992)]

$$\begin{aligned} x^i &= \tfrac{F_c x^c}{P_x z^c} \\ y^i &= \tfrac{F_c y^c}{P_y z^c}, \end{aligned} \tag{3.22}$$

where F_c is the focal length of image sensor, P_x and P_y are inter-pixel spacing along x and y axis on the image plane.

Another way to get the relationship between camera frame and object frame is using homogeneous transformation matrix based on Euler angle representation. However, it may inevitably cause singularity due to implementation of rotation matrix. If that is the case,

the columns in homogeneous transformation matrix would be linear dependent, i.e. there may exists a feature point on the surface of the object excluding the one coincident with the origin in camera frame, being projected to the origin of the camera frame. It is obviously a incorrect coordinate transformation and will make Eq. (3.22) become $\frac{0}{0}$ which is invalid for solving image coordinate (x^i, y^i). Fortunately, dual quaternion avoids the occurrence of matrix singularity by using a compact form in Eq. (3.19) that successfully prevents the possible incorrect coordinate transformation.

As shown in Eqs. (3.20)-(3.22), the coordinates on image plane is derived with respect to the dual quaternion related state. Correspondingly, the true coordinates of feature point on image plan are captured by digital cameras, which are handled as the measurement, denoted as $\mathbf{z}$. In order to more precisely estimate the spatial rigid motion of the concerned object, we track the projection of three feature points on the image plane. In addition, we put unit constraint on the real part of dual quaternion such that $\mathbf{q_r}^T\mathbf{q_r} = 1$, as well as orthogonal constraint, $\mathbf{q_r}^T\mathbf{q_d} = 0$. Under these assumptions, the components in the measurements are composed of

$$\mathbf{z} = \begin{bmatrix} x_1^i & y_1^i & x_2^i & y_2^i & x_3^i & y_3^i & \mathbf{q_r}^T\mathbf{q_d} & \mathbf{q_r}^T\mathbf{q_r} \end{bmatrix}^T . \tag{3.23}$$

According to findings in Eq. (3.22), the measurements are functions of states $\mathbf{x}$, labeled as $h(\mathbf{x})$. We then can find the discrete observation matrix H^t in Eq. (3.11) according to $H^t = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \mid_{\mathbf{x}=\mathbf{x}^{t-}}$.

By now, we have obtained the state transition and observation models for estimation of spatial rigid motion using a single image sensor. By applying the above described EKF algorithm, we can process the observation data based on the developed models. However, due to limited field-of-view of single sensor, the tracking data of feature points may not be available during the observation procedure. Therefore, a multi-sensor observation framework is developed to improve the estimation performance.

4.4 Position Tracking and State Estimation Based on EKF in Multi-Agent System

4.4.1 Multi-Sensor Multi-Agent Networks and Problem Formulation

As we discussed in the introduction section, the disadvantage of using single image sensor is that when the feature points move of out the view zone, the measurement data will not be available. The limitation of single image sensor will significantly impacts the estimation accuracy. Therefore, we propose a multi-sensor network to track the object's spatial motion from different sensors simultaneously.

In a connected sensor network as shown in Fig. 4.2, we assume each sensor can communicate with its neighbors. There is no central processor and the network is not fully connected. However, as long as there is connection which is defined by the entries of the adjacency matrix $\mathcal{A}$ between any two nodes in the network, the two connected sensors can communicate with each other. Furthermore, they are able to spread the information among the connected network finally. In such system, the information filter or some other data fusion algorithm that requires fully connected network cannot be applied in partially connected system. In addition, the fully connected network requires large data communication and storage which may bring difficulty for implementation with the increase of data numbers. Furthermore, without consensus constraints, there is no limitation to converge the final result to the average consensus. For a network system with N sensors described in Fig. 4.2, adjacency matrix $\mathcal{A}$ which is a symmetric matrix with zero diagonal entries indicates the neighbors for sensor i by the off diagonal entries $\mathcal{A}_{i,j}$, $i, j = 1, \ldots, N$, $i \neq j$. If $\mathcal{A}_{i,j} = 1$, then agent i can communicate with agent j. It is expected that the estimates obtained from agent i, $\hat{\mathbf{x}}_i$, to be identical to $\hat{\mathbf{x}}_j$ as well as the other estimates from the connected neighbors. By passing the identity request from one node to the other in the network, we can transfer the consensus request in the system. With the neighborhood consensus constraints on the

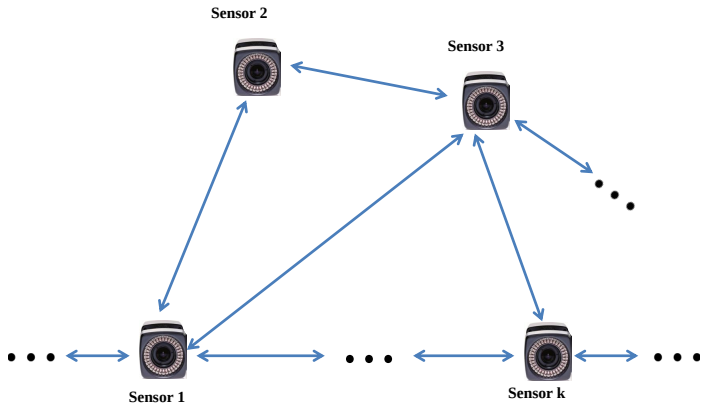

Figure 4.2 Communication and framework of a multi-sensor multi-agent network (Agent is embedded with sensor.)

estimates, we have the following relationships:

$$a_{i,j}\hat{\mathbf{x}}_i - a_{i,j}\hat{\mathbf{x}}_j = 0, \quad i = 1, \ldots, N, \quad i > j, \tag{4.24}$$

where $a_{i,j}$ denotes the element $\mathcal{A}_{i,j}$ in matrix $\mathcal{A}$. If there is a connection between agent i and j, the consensus condition expressed in Eq. (4.24) will have $\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_j$, otherwise such consensus constraint between node i and j does not exist. Since matrix $\mathcal{A}$ is symmetric, we will have $a_{i,j} = a_{j,i}$. As summary, the multi-sensor network estimation problem with consensus constraints at time step k is formulated as

$$\begin{aligned} min. \quad & \Sigma_{i=1}^{N}(\mathbf{x}_i^t - \mathbf{x}_i^{t-})^T(P_i^{t-})^{-1}(\mathbf{x}_i^t - \mathbf{x}_i^{t-}) + (\mathbf{z}_i^t - h_i(\mathbf{x}_i^t))^T R^{-1}(\mathbf{z}_i^t - h_i(\mathbf{x}_k^t)) \\ s.t. \quad & a_{i,j}\mathbf{x}_i^t - a_{i,j}\mathbf{x}_j^t = 0,\ i = 1, \ldots, N,\ i > j, \end{aligned} \tag{4.25}$$

where $\mathbf{x}_i^t$ denotes the state vector in (3.15) estimated from agent i at time instance t and N is the number of sensors in the connected network.

4.4.2 Solving with Dual Decomposition and Subgradient Method

Based on Eq.(4.25), we introduce Lagrangian multipliers $\boldsymbol{\lambda}$ for the additional equality constraints in EKF objective function so that it becomes

$$\begin{aligned} L &= \Sigma_{i=1}^{N}[(\mathbf{x}_i^t - \mathbf{x}_i^{t-})^T (P_i^{t-})^{-1}(\mathbf{x}_i^t - \mathbf{x}_i^{t-}) \\ &\quad +(\mathbf{z}_i^t - h_i(\mathbf{x}_i^t))^T R^{-1}(\mathbf{z}_i^t - h_i(\mathbf{x}_i^t)) + \sum_{j=1}^{i} \boldsymbol{\lambda}_{i,j}^T (a_{i,j}\mathbf{x}_i^t - a_{i,j}\mathbf{x}_j^t)]. \end{aligned} \tag{4.26}$$

Obviously, the above Lagrangian consists of N subproblems and can be solved by

$$\begin{aligned} L_i &= \min_{\mathbf{x}_i^t}[(\mathbf{x}_i^t - \mathbf{x}_i^{t-})^T (P_i^t)^{-1}(\mathbf{x}_i^t - \mathbf{x}_i^{t-}) \\ &\quad +(\mathbf{z}_i^t - h_i(\mathbf{x}_i^t))^T R_i^{-1}(\mathbf{z}_i^t - h_i(\mathbf{x}_i^t)) \\ &\quad +(\sum_{j=1}^{i-1} \boldsymbol{\lambda}_{i,j}^T a_{i,j} - \sum_{j=i+1}^{N} \boldsymbol{\lambda}_{j,i}^T a_{j,i})\mathbf{x}_i^t], \quad i = 2, \ldots, N-1. \end{aligned} \tag{4.27}$$

together with

$$\begin{aligned} L_1 &= \min_{\mathbf{x}_1^t}[(\mathbf{x}_1^t - \mathbf{x}_1^{t-})^T (P_1^t))^{-1}(\mathbf{x}_1^t - \mathbf{x}_1^{t-}) \\ &\quad +(\mathbf{z}_1^t - h_1(\mathbf{x}_1^t))^T R_1^{-1}(\mathbf{z}_1^t - h_k(\mathbf{x}_1^t)) \\ &\quad - \sum_{j=i+1}^{N} \boldsymbol{\lambda}_{j,i}^T a_{j,i}\mathbf{x}_1^t], \end{aligned} \tag{4.28}$$

and

$$\begin{aligned} L_N &= \min_{\mathbf{x}_N^t}[(\mathbf{x}_N^t - \mathbf{x}_N^{t-})^T (P_N^t)^{-1}(\mathbf{x}_N^t - \mathbf{x}_N^{t-}) \\ &\quad +(\mathbf{z}_N^t - h_N(\mathbf{x}_N^t))^T R_N^{-1}(\mathbf{z}_N^t - h_N(\mathbf{x}_N^t)) \\ &\quad + \sum_{j=1}^{i-1} \boldsymbol{\lambda}_{i,j}^T a_{i,j}\mathbf{x}_N^t]. \end{aligned} \tag{4.29}$$

These subproblems are independent with each other and they can be solved individually. Therefore, we pay more attention on each Lagrangian subfunctions instead of the summation problem.

Consider $H_i^t = \frac{\partial h_i(\mathbf{x_i})}{\partial \mathbf{x_i}} \mid_{\mathbf{x_i}=\mathbf{x}_i^{t-}}$ and use the first order optimality condition to determine $\hat{\mathbf{x}}_i^t$. We take the derivative of L_i with respect to $\mathbf{x}_i^t$ to give

$$\begin{aligned}\frac{dL_i}{d\mathbf{x}_i^t} &= (P_i^t)^{-1}(\mathbf{x}_i^t - \mathbf{x}_i^{t-}) - (H_i^t))^T R_i^{-1}(\mathbf{z}_i^t - h_i(\mathbf{x}_i^{t-})) \\ &\quad + \sum_{j=1}^{i-1} \boldsymbol{\lambda}_{i,j} a_{i,j} - \sum_{j=i+1}^{N} \boldsymbol{\lambda}_{j,i} a_{j,i},\ i = 1, \ldots, N. \end{aligned} \tag{4.30}$$

Lagrangian function will achieve the minimum value if Eq. (4.30) is zero. Hence, we get the updated function of $\hat{\mathbf{x}}_i^t$ at time interval k in the form of

$$\hat{\mathbf{x}}_i^t = \hat{\mathbf{x}}_i^{t-} + P_i^t H_i^{tT}(R_i^t)^{-1}(\mathbf{z}_i^t - h_i(\hat{\mathbf{x}}_i^{t-})) - P_i^t(\sum_{j=1}^{i-1} \boldsymbol{\lambda}_{i,j} a_{i,j} - \sum_{j=i+1}^{N} \boldsymbol{\lambda}_{j,i} a_{j,i}). \tag{4.31}$$

In the above equation $P_i^t H_i^{tT}(R_i^t)^{-1} = K_i^t$ is the Kalman gain for agent i which is similar to the Kalman gain update in EKF algorithm. The only difference between them is the last additional term. The extra term can be handled as adjustment of estimates to satisfy consensus constraints. By using subgradient method, Lagrangian multipliers are updated by

$$\boldsymbol{\lambda}_{i,j}^{q+1} = \boldsymbol{\lambda}_{i,j}^q + \alpha^q a_{i,j}(\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_j^t), \quad i = 1, \ldots, N, \quad j = 1, \ldots, i-1, \tag{4.32}$$

where α^q is the step size to control the speed of the adjustment at qth iteration. In this estimation problem, the step size is set as $\alpha^q = r(\beta)^q$, where $r \in \mathbb{R}$ and $\beta \in (0, 1]$ are given constants.

However, since Euler angle representation needs triangular function which is well known to be a periodic function, the inverse of that may cause relatively large estimation error under the fast rotation condition. Although Eq. (4.32) still works well due to the unvarying difference of two estimates $\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_i^t$, the estimates calculated by Eq. (4.31) converge to consensus, but with relatively large estimation error due to the large bias of one step estimation $\mathbf{x}_i^-$. While the state estimation using quaternion representation can be obtained directly without any periodic function or inverse operation.

4.4.3 Solving with Distributed Newton Method

The above consensus based distributed estimation approach using dual subgradient method requires iterative coordination between networked sensors. For each iteration, the individual sensor must process the new estimates with the updated coordination variables using EKF. Even though the dual quaternion based models simplified the representation of spatial rigid motion, at least eight dual elements are included in the estimates for every iteration, so this process is time and resource consuming. It is imperative to develop a faster convergent distributed estimation algorithm that will lead to minimum estimation error with less iterative coordination. Inspired by the rapid convergence of the Newton's method in solving network utility maximization problems [Dolev et al. (2009); Wei et al. (2010)], we propose a Newton-type distributed estimation algorithm, aiming at increasing the rate of convergence.

Since the objective function is decomposable in terms of x_i, Eq.(6.27) and (6.28) can be expressed locally as

$$\Delta \mathbf{x}_i^q = -(\nabla^2 f_i^q)^{-1}((\nabla f_i^q) + C_i^T w_i^q) \tag{4.33}$$

$$(C_i(\nabla^2 f_i^q)^{-1} C_i^T)\mathbf{w}_i^q = -C_i(\nabla^2 f_i^q)^{-1}\nabla f_i^q. \tag{4.34}$$

The diagonal block matrices in Hessian can be calculated individually at iteration q, denoted as $\nabla^2 f_i^q = \frac{\partial^2 f_i}{\partial \mathbf{x}_i^2}|_{\mathbf{x}_i=\mathbf{x}_i^q}$. The estimation problem with consensus constraints formulated in (4.25) can be handled as one of the general constrained optimization problems. We use $\nabla f_{\mathbf{x}(i)}^q$ and $\nabla^2 f_{\mathbf{x}(i)}^q$ to represent the elements in the gradient vector and Hessian corresponding to $\mathbf{x}(i)$ at iteration q. The Hessian and gradient of the objective function are stated as following,

$$\nabla f_i^q = (P_i^-)^{-1}(\mathbf{x}_i^q - \mathbf{x}_i^{q-}) - H_k(i)^T R^{-1}(\mathbf{z}_i - h_i(\mathbf{x}_i^q)) \tag{4.35}$$

$$\nabla^2 f_i^q = (P_i^-)^{-1} + H_i^T R_i^{-1} H_i, \tag{4.36}$$

where $\mathbf{x}_i^q$ denotes state vector in (3.15) estimated from agent i at iteration q and time interval t. For notation simplification, we ignore the time index t in superscript of above

expressions. After calculating ∇f_i^q and $\nabla^2 f_i^q$ and substituting them in (4.36) and (4.35), ω_i and local Newton direction Δ_i can be updated by (4.33) and (4.34)

4.5 Simulation and Discussion

Two simulation scenarios are provided in this section, one is general motion tracking without considering dynamics and the other one is aircraft motion tracking with integrated dynamics model. In both cases, estimates from distributed Newton method is demonstrated and compared with those obtained from dual subgradient and individual sensors.

4.5.1 A General Motion Estimation Case

Figure 4.3 demonstrates the layout of image sensors and trajectories of the tracking points on the moving object. To make it simple, we use two image sensors which can communicate with each other. Sensor 1 is set 10 meters directly below the object initial position. Sensor 2 is set at the left hand side of the object initial position with 10 meters shifting along $-y^o$ axis and a rotation of 90^o with respect to $-x^o$ axis. Three points on surface of the object are selected as feature points. They are located on the axis of object frame and are one meter away from the origin. The object is moving along positive x^o axis with constant linear velocity $v_x = 1m/s$. Simultaneously, it is rotating around axis x_0 with a constant angular velocity $\omega_x = 2\pi\, rad/s$. Table 5.2 shows relative parameters of the observing model. The measurement noise considered when observing the projections in the image plane is Gaussian white noise with $0.06\, pixel^2$ variance.

Table 4.1 Parameter Settings

Pixel spacing along x, P_x	$2 \times 10^{-5}\, m/pixel$
Pixel spacing along y, P_y	$2 \times 10^{-5}\, m/pixel$
Sample period, T	$0.02\, s$
Focal length, F_c	$0.02\, m$
Measurement noise variance	$0.06\, pixel^2$

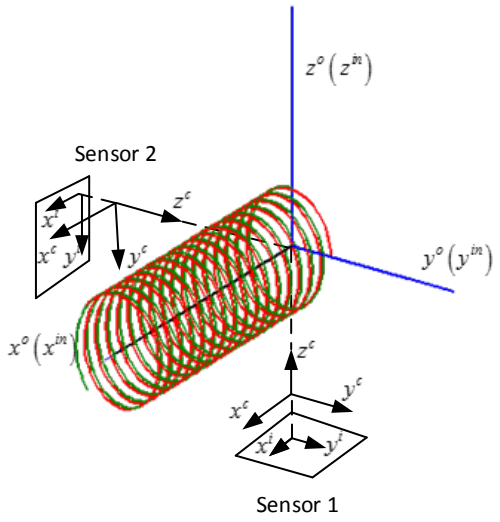

Figure 4.3 Demonstration of sensor locations and object movement.

There are two important parameters involved in EKF, process noise covariance matrix Q and measurement noise covariance matrix R. In our model, the object in the simulation has constant linear and angular velocity. It is expected that there is no dramatic change in the system states. Therefore, it is reasonable to use a constant matrix Q, which will not significantly affect the tracking accuracy. In the simulation, Q is set as a diagonal matrix. The diagonal elements corresponding to dual quaternion states in Q are set as 0.01. The elements related to angular and linear velocity are set as 0.1. Another parameter is the measurement noise covariance matrix R. Since we consider white noise in the observation model, R is assumed to be a diagonal matrix with 0.06 in diagonal elements, as shown in Table 5.2.

Before we proceed with EKF, two parameters, the initial states and the covariance matrix of the estimation error, need to be initialized at the beginning. To demonstrate the effectiveness of the proposed distributed estimation algorithm, offsets are added to real initial states. For example, the elements in initial position vector have offset of ± 0.1 meters, the elements in linear and angular velocity vectors have offset of -1 and -2π, respectively. Correspondingly, we set up the initial covariance matrix of the estimation error, P_0^-, as a

diagonal matrix according to the definition, $P_0^- = E[(x_0 - x_0^-)(x_0 - x_0^-)^T]$. It is expected that even with the large offsets of initial states, the proposed distributed estimation algorithm can eventually converge to the real states.

Based on the above simulation scenario, three simulation examples are given below. The first implements EKF to estimate the time history of dual quaternions, angular and linear velocities using individual image sensors, where no cooperation is considered between the sensors. The second uses dual subgradient-based distributed estimation algorithm where cooperation between the connected two sensors are considered. The third applies the proposed Newton-type distributed estimation algorithm.

Simulation results using Newton-type distributed method to estimate the spatial motion are given below. The estimates of dual quaternion elements are provided in Figs.4.4 and 4.5. The estimates of angular velocity and translational velocity are provided in Figs.4.6 and 4.7, respectively. In addition, Fig.4.8 demonstreates the magnified part of ω_x obtained from three methods to verify that the estimates from the Newton-type distributed method converge to the real values much faster than results from individual sensors or from the dual subgradient method. The individual sensor will yield different estimates. By using dual subgradient method, the estimates obtained from two sensors will eventually converge to consensus. However, the estimation accuracy will be affected by the single sensor which generates low precision estimates. Consequently, it takes much longer time to converge to consensus when compared to the Newton-type distributed method.

To save space, the estimation plots from individual sensors and dual subgradient method are not provided here. Instead, we provide the comparison of average mean square error (MSE) from individual sensors, dual subgradient, and Newton methods in Table II. It demonstrates a reduced MSE in pose and position estimation using Newton-type method when compared with a single EKF or dual subgradient method. Especially for the linear velocity along y direction, estimation accuracy from Newton-type method is much higher than other methods. Meanwhile, Newton-type method provides two strictly identical esti-

mates, which satisfies the exact consensus constraints. Therefore, even though the position or velocity of the tracked object is unknown, the Newton-type distributed algorithm provides a reliable estimation under the condition where there are different type of sensors with different level of accuracy. Moreover, when increasing the number of sensors, the reliability of the estimates given by distributed Newton method will be enhanced.

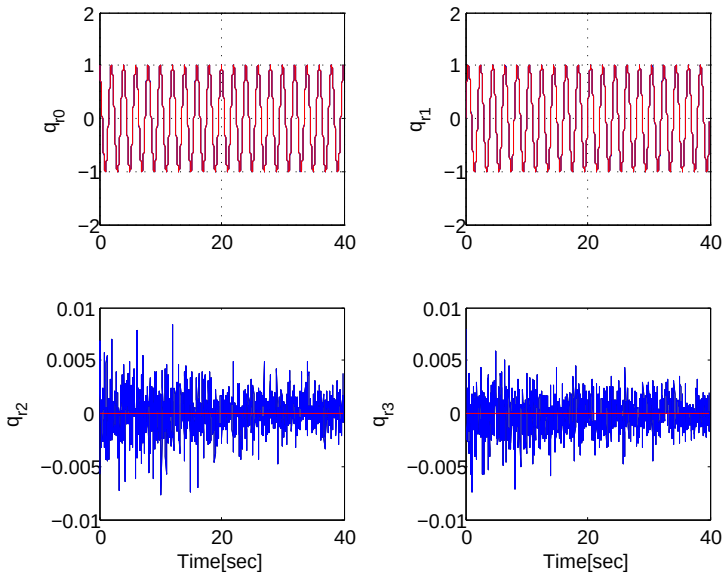

Figure 4.4 Comparison of real (red) and estimated (blue) real part elements of dual quaternions from cooperative sensors based on Newton-type distributed algorithm.

4.5.2 Controlled General Rigid Body Motion Estimation Case

Based on the improved performance of Newton-type distributed method in general motion estimation as verified above. It is now implemented in motion estimation of a general 3-D rigid body. Sensor network is demonstrated in Fig.4.9 where eight cameras are installed to cover the whole observation region which is demonstrated as the cuboid formed by line segments connecting neighbor sensors. In additional to the parameters listed in Table 5.2, the angle of view for each cameras is 94^o. Furthermore, the minimum pixel number for a point to be observed on image plane is 4, i.e. the feature point on object surface is measured

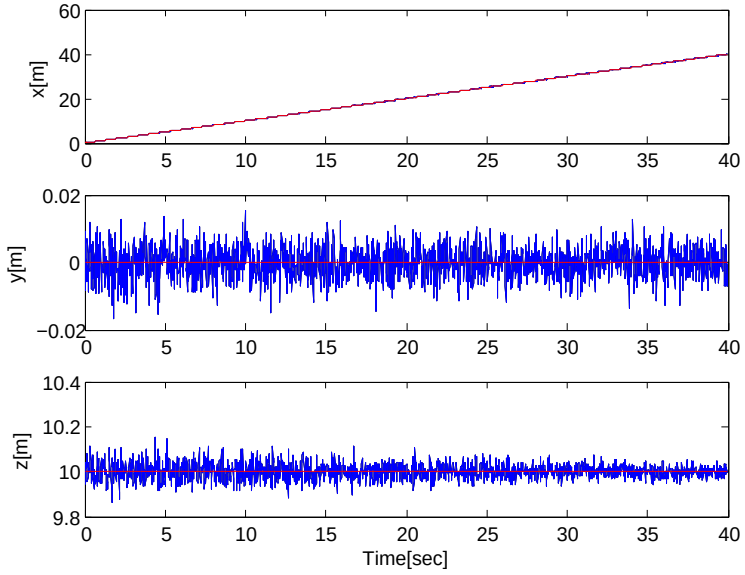

Figure 4.5 Comparison of real (red) and estimated (blue) trajectory of motion from cooperative sensors based on Newton-type distributed algorithm.

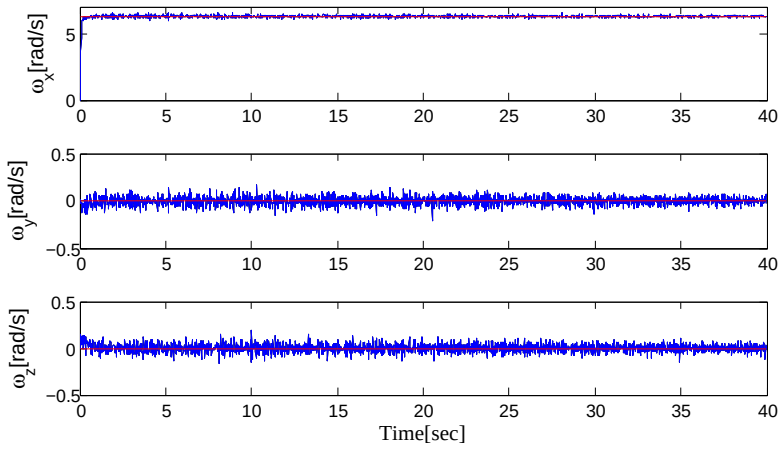

Figure 4.6 Comparison of real (red) and estimated (blue) angular velocity from cooperative sensors based on Newton-type distributed algorithm.

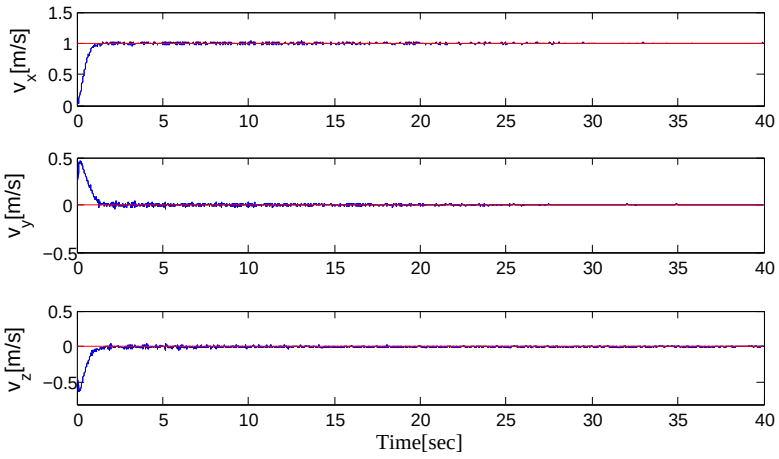

Figure 4.7 Comparison of real (red) and estimated (blue) translational velocity from co-operative sensors based on Newton-type distributed algorithm.

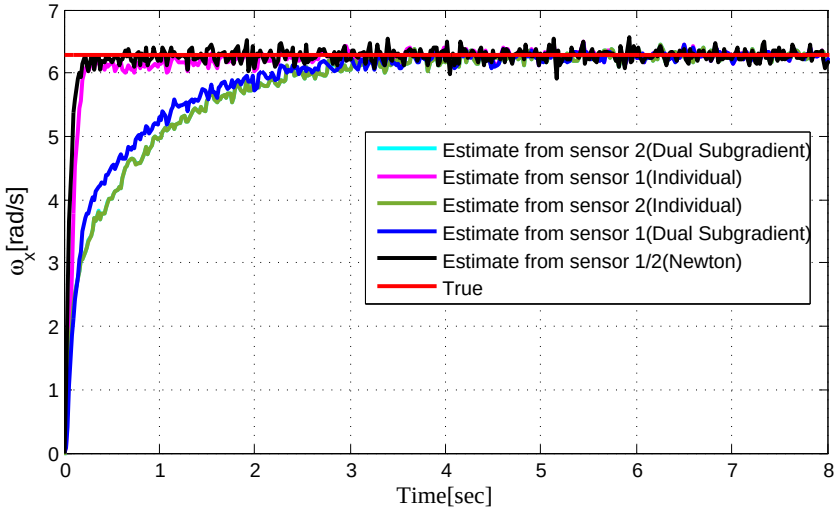

Figure 4.8 The magnified part of ω_x from three estimation methods.

Table 4.2 Comparison of MSE from two individual sensors (EKF), Dual Subgradient (DS) and Newton-Type (Newton) methods.

State Variables	MSE (Sensor1 EKF)	MSE (Sensor2 EKF)	MSE (Sensor1 DS)	MSE (Sensor2 DS)	MSE (Newton)
q_{r0}	2.66×10^{-6}	2.69×10^{-6}	2.46×10^{-6}	2.64×10^{-6}	1.04×10^{-6}
q_{r1}	3.77×10^{-6}	3.26×10^{-6}	5.44×10^{-6}	3.20×10^{-6}	1.14×10^{-6}
q_{r2}	2.25×10^{-6}	1.49×10^{-6}	1.92×10^{-6}	1.47×10^{-6}	0.48×10^{-6}
q_{r3}	1.48×10^{-6}	1.58×10^{-6}	1.45×10^{-6}	1.57×10^{-6}	0.52×10^{-6}
x	4.78×10^{-3}	5.15×10^{-3}	4.51×10^{-3}	5.02×10^{-3}	3.64×10^{-3}
y	4.71×10^{-5}	1.23×10^{-3}	4.63×10^{-5}	1.21×10^{-3}	2.02×10^{-5}
z	2.16×10^{-3}	1.57×10^{-3}	4.47×10^{-3}	1.53×10^{-3}	1.20×10^{-3}
ω_x	0.0813	0.2102	0.2119	0.1860	0.0524
ω_y	0.0032	0.0028	0.0024	0.0024	0.0021
ω_z	0.0021	0.0172	0.0107	0.0116	0.0021
v_x	0.0212	0.0132	0.0019	0.0028	0.0019
v_y	0.0281	4.2697	3.5221	3.5024	0.0025
v_z	0.0059	0.0157	0.0062	0.0064	0.0035

within a distance, which can be calculated through pinhole projection formula associated with the size of feature point. Knowing that feature point is a colored circle with diameter of $0.1m$, then the maximum measured distance from feature point to the image plane is $16.686m$.

Table 4.3 shows the initial and final position, orientation and velocity information of the rigid body, as well as the relative transformation of each camera with respect to camera 1, whose camera frame is handled as the reference frame coincidence with inertial frame. We assume each camera is fixed during the entire observation interval. Moreover, the rigid body is maneuvered to reach the final position and attitude from the specified initial states. To guarantee that the maneuver control is accomplished within several seconds, we applied PID control law and adjust parameters to reduce the time required for the rigid body to achieve stable states. Figure 4.9 illustrates an observation region defined by a multi-sensor network. As the controlled rigid body moves in the observation region, partial of the cameras

may not able to observe all feature points on the surface of the target due to field-of-view constraints, which leads to a time-varying network topology illustrated in Fig. 4.10. The varying topology at different time interval is also shown in Fig. 4.10. When any camera, indexed by a number 1-8, cannot observe all future points, it is assumed to be disconnected from the network since it is not providing measurements at the moment. The remaining available cameras automatically form a circular pattern using wireless connection between two adjacent vertices, denoted as dash links in Fig. 4.10.

By using the distributed Newton-type method, estimation results in terms of the position, orientation, and velocity of the rigid body in the body frame are shown in Figs.4.11-4.14. Red curve indicates the real value and black curve indicates the estimated results from sensor 1/2.

Table 4.3 Layout of the UAV and Camera 2 w.r.t Camera 1

	Position Coordinate/m	Attitude (real part)
Initial Rigid Body	[12 7 15]	[0.4286 0.7047 $-$ 0.3618 0.4345]
Final Rigid Body	[0 0 0]	[1 0 0 0]
Camera 2	[0 0 16]	[0.5 0.5 0 0]
Camera 3	[10 0 0]	[0 0 0 0]
Camera 4	[10 0 16]	[0.5 0.5 0 0]
Camera 5	[0 10 0]	[0 0 0 0]
Camera 6	[0 10 16]	[0.5 0.5 0 0]
Camera 7	[10 10 0]	[0 0 0 0]
Camera 8	[10 10 16]	[0.5 0.5 0 0]

Table 4.4 A general 3-D rigid body Motion Esimation MSE

state	x	y	z
Trajectory	0.0002	0.0036	0.0002
Angular Velocity	0.0015	0.0014	0.0010
Translational Velocity	0.0468	0.0636	0.0476

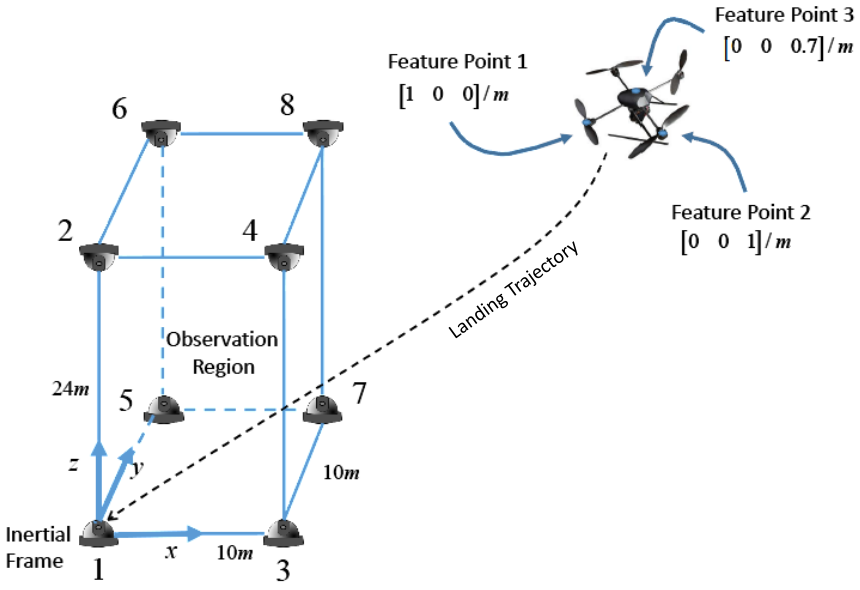

Figure 4.9 A general 3-D rigid body motion estimation layout

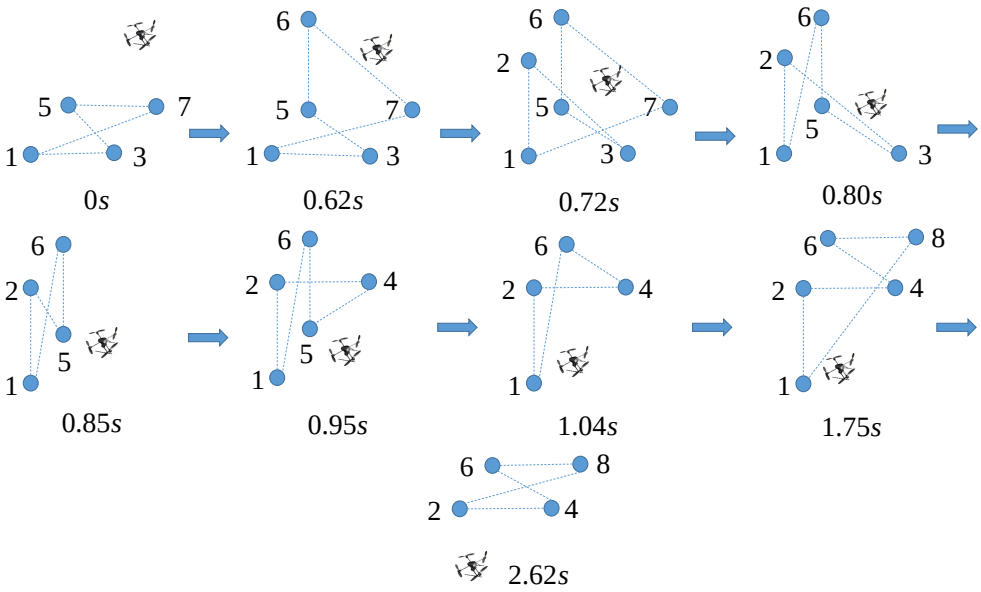

Figure 4.10 Sensor network topology at different time intervals

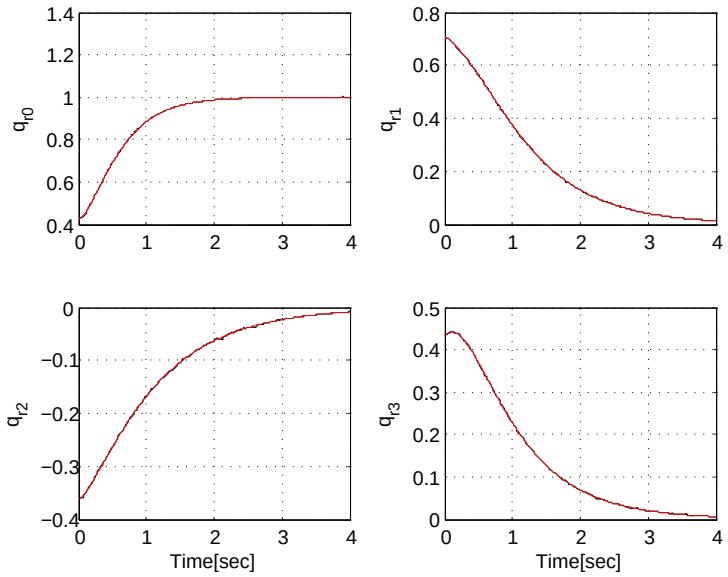

Figure 4.11 A general 3-D rigid body real (red) and estimated (black) real part elements of dual quaternions from cooperative sensors based on Newton-type distributed algorithm.

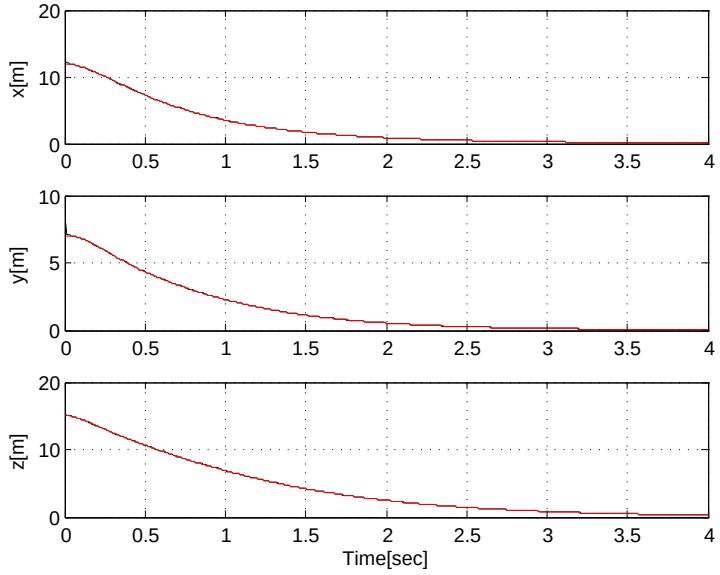

Figure 4.12 A general 3-D rigid body real (red) and estimated (black) trajectory of motion from cooperative sensors based on Newton-type distributed algorithm.

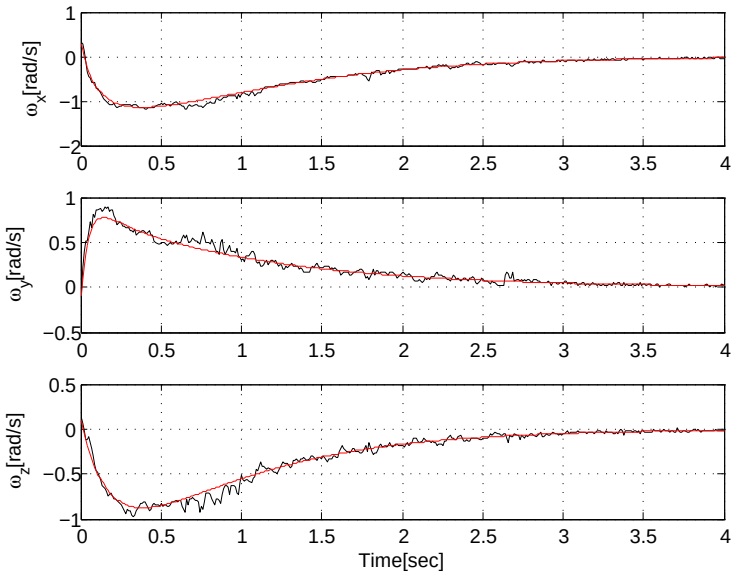

Figure 4.13 A general 3-D rigid body real (red) and estimated (black) angular velocity from cooperative sensors based on Newton-type distributed algorithm.

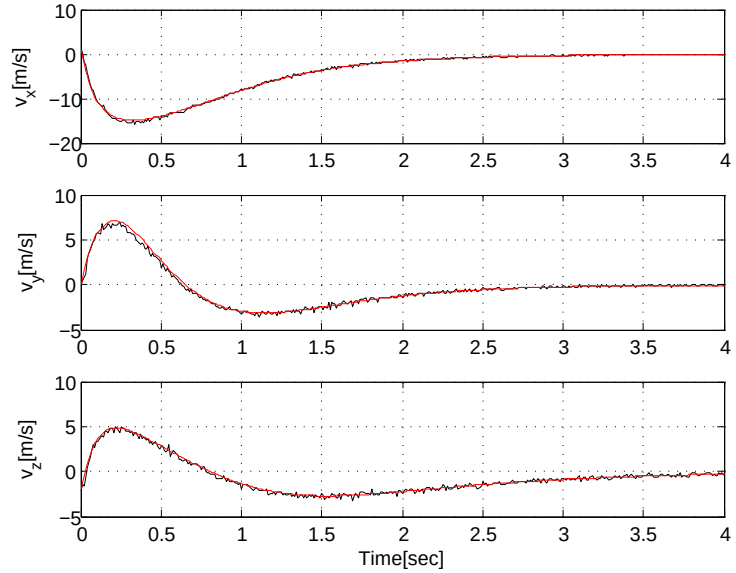

Figure 4.14 A general 3-D rigid body real (red) and estimated (black) translational velocity from cooperative sensors based on Newton-type distributed algorithm.

4.6 Conclusion

We models spatial rigid motion using dual quaternions. Based on this model, extended Kalman Filter is implemented to estimate spatial motion in real-time. In particular, we track the projection of object feature points on image plane and use the two dimensional tracking data to estimate the motion with six-degree-of-freedom. To avoid the impact of the limitation of single sensor and improve the system fault tolerance, dual subgradient distributed estimation algorithm is proposed based on multi-sensor multi-agent network. Due to the low convergence rate and the low precision of estimation, a new distributed estimation approach based on distributed Newton method is proposed. The improved computational efficiency and estimation performance using the integrated dual quaternion modeling and distributed estimation framework is verified by simulation examples.

CHAPTER 5. DISTRIBUTED OPTIMIZER DESIGN: DISTRIBUTED PATH PLANNING FOR BUILDING EVACUATION GUIDANCE

5.1 Problem Statement

In-building evacuation requires evacuees to be guided to the available exits with the minimum time spent when hazard happens. We group evacuees in terms of their locations and assume there is one leader in each group. Each leader has a smart phone or personal digital assistant (PDA) that can communicate with neighboring evacuees within certain distance to form a evacuation group, locate position, and retrieve hazard spreading information from all sensors. The problem is for each group to design individual evacuation path by using the PDA owned by the leader. And the final decision will be shared among group members at the same location. The optimal path planning is supposed to be the shortest/evacuation time minimized. Meanwhile, by following the optimal evacuation path, each group is able to successfully avoid the possible congestion and hazard happened in corridor or the area close to the exit.

5.2 Building Evacuation and Hazard Spreading Model

Before constructing the evacuation and hazard spreading models, assumptions associated with the sensor layout, number of evacuees, social behaviors, and edge cost are stated first.

Assumption 5.2.1. *At least one sensor used for detecting potential hazard, such as fire or explosion, is installed in each room of the object building. Same type of sensors are installed*

in the corridor with an equivalent distance interval. All of them can report the occurrence of hazard and predict the hazard arriving time. There are communication devices in each corridor and exit, which are responsible for communicating and coordinating with distributed computation nodes.

Assumption 5.2.2. *Evacuees are randomly scattered in the building. There are capacity constraints for each corridor and exit.*

Assumption 5.2.3. *Individual behavior is affected by group members [Crano (2000)]. For example, people prefer to follow others rather than find alternative routes during emergent evacuation if they are not familiar with the building layout.*

Assumption 5.2.4. *Once a hazard area is detected, there is a delay before all of the edges involved with the corresponding hazard area become unavailable. After the delay period, the cost of unavailable edges will be assigned by an infinite number.*

The evacuation graph includes essential information, such as graph topology, edge cost, and hazard spreading consequence along all edges. According to assumption 5.2.1, two neighboring sensors are directly connected if there is a door between them or they are adjacent in the corridor. The sensors represent vertices in a evacuation graph and an edge exists between two connected sensors. Edge cost in the evacuation graph is evaluated by the time consumption for evacuees to traverse the relative edge. The initial edge cost is predetermined from off-line investigation and experiment according to the edge length, individual group size, and moving speed. A simple example of the evacuation graph is shown in Fig. 5.1.

The hazard spreading model estimates the time required for a hazard area spreading from one sensor to its neighboring ones. Even though there is a barrier, like a wall between two sensors, the hazard, such as fire, can propagate through them after a relatively long duration. Hence, different from the evacuation graph which assumes no connections when blocked by barriers, the hazard spreading graph includes more edges. The relative edge cost

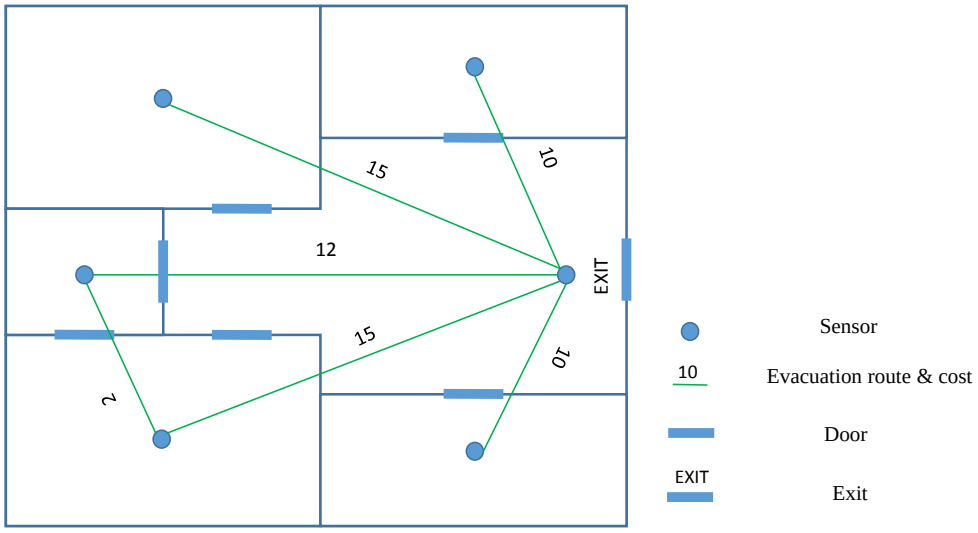

Figure 5.1 An example of evacuation graph

is computed by simulating virtual fire or smoke spreading models to evaluate hazard effect on evacuees [Gorbett et al. (2008)]. Additional parameters, such as lining materials, structural dimensions, ventilation and etc., are required in the simulation. The corresponding hazard spreading graph for the scenario depicted in Fig. 5.1 is demonstrated in Fig. 5.2.

The evacuation and hazard spreading graphs can be represented by weighted adjacency matrices, denoted as $A^i(t)$ for evacuee group i and $A^H(t)$ for hazard spreading, respectively. Both of them are symmetric matrices with zero diagonal entries and non-zero off diagonal entries, denoted as $A^i_{u,v}(t)$ and $A^H_{u,v}(t)$, $u, v = 1, \ldots, N$, $u \neq v$, where u and v are graph vertices, $(u, v) \in \mathcal{G}$ denotes the edge connecting vertices u and v, and $\mathcal{G}$ represent the graph. If there is no available edge between vertices u and v at time instant t, $A^i_{u,v}(t), A^H_{u,v}(t) \to \infty$. Time consuming of hazard spreading from one spot to another is predicted through simulation model, such as FSSIM [Floyd et al. (2005)], FDS [Gawad and Ghulman (2015)] and so on. Specifically, based on computational fluid dynamics technique, fire and smoke spreading dynamics are pre-determined once the building layout and structure is known. Therefore, without lose of generality, hazard spreading matrix remains constant, i.e. $A^H(t) = A^H$.

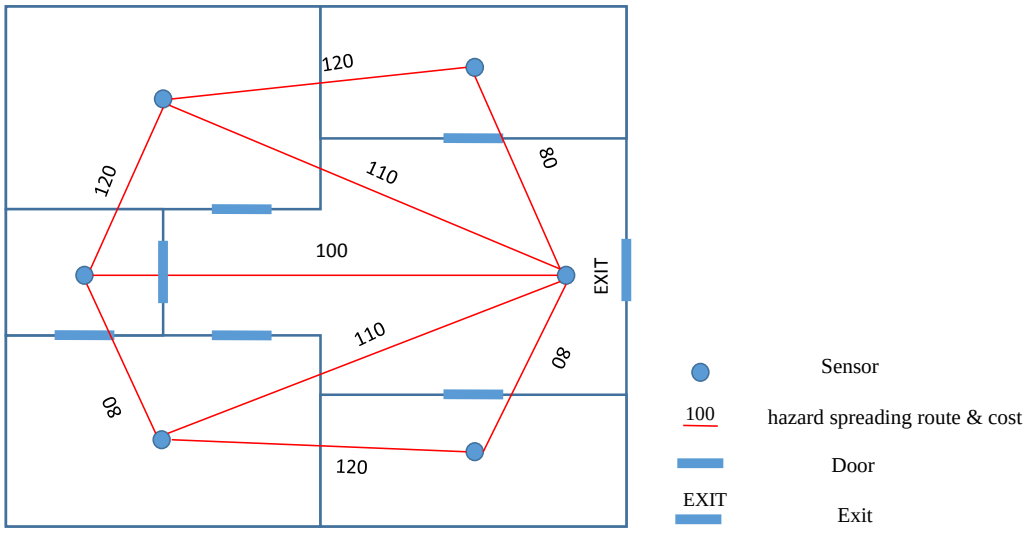

Figure 5.2 An example of hazard spreading graph

However, when a great number of evacuees trying to reach the safe region through the same path simultaneously, congestion along certain edges will slow down the moving speed of the evacuee flow and consequently increases the corresponding edge cost. $A^i(t)$ is time-varying due to the dynamic hazard effect and capacity constraints. According to the dimension of the object building, two parameters are considered. One is the threshold of evacuee number, n_l, before it causes increment of the edge cost and the other one is the edge capacity n_c that reduces moving speed to shuffling status without allowing cross or reverse movement. When considering the congestion effect and hazard spreading, edge cost is determined by the number of evacuees, denoted by $n_{u,v}(t)$, along edge (u, v) at time interval t.

Thus, the edge cost $A^i_{u,v}(t), i = 1, \ldots, M$, is updated by the following expressions, where edge cost is related to number of evacuees and hazard spreading time.

$$A^i_{u,v}(t) = \begin{cases} c^i_{u,v}, & if \quad n_{u,v}(t) \le n_l \\ & \quad and \quad t < \min\{s_u + d_u, s_v + d_v\} \\ f^i_{u,v}(n_{u,v}(t)), & if \quad n_l < n_{u,v}(t) \le n_c \\ & \quad and \quad t < \min\{s_u + d_u, s_v + d_v\} \\ F^i_{u,v}(n_{u,v}(t)), & if \quad n_{u,v}(t) > n_c \\ & \quad and \quad t < \min\{s_u + d_u, s_v + d_v\} \\ \infty & if \quad t \ge \min\{s_u + d_u, s_v + d_v\} \end{cases} \tag{2.1}$$

where $c^i_{u,v}$ is the cost for group i traverse edge (u, v) and M is the number of evacuee groups. Functions $f^i_{u,v}(n_{u,v}(t))$ and $F^i_{u,v}(n_{u,v}(t))$ are determined by the walkway service level and path width [Kisko et al. (1998)] according to user's guide of NVACNET4 [Lu et al. (2005)]. For example, time consumed for evacuee groups with different number of members passing through corridors on the second floor of Howe Hall at Iowa State University is determined by

$$f^i_{u,v}(n_{u,v}(t)) = \begin{cases} 3.1 \sim 3.4 & if \quad 3 < n_{u,v}(t) \le 6 \\ 3.4 \sim 3.9 & if \quad 7 \le n_{u,v}(t) \le 9 \end{cases}. \tag{2.2}$$

If $n_{u,v}(t) > 9$, one has

$$F^i_{u,v}(n_{u,v}(t)) = [3.9 + w(n_{u,v}(t) - 10)] \sim [7.5 + w(n_{u,v}(t) - 10)], \tag{2.3}$$

where w denotes the degree of decreased moving speed when additional evacuees move into edge (u, v). Equation (2.2) indicates the lower bound $n_l = 3$ and edge capacity $n_c = 9$. Therefore, a few additional evacuees could significantly increase the time required for a evacuee group moving through edge (u, v). To analyze the hazard effect on the edge cost of the evacuation graph, the hazard spreading paths are generated along available edges in the hazard spreading graph once a specific hazard spot is detected. The spreading time, s_u, for the detected hazard to arrive at vertex u is determined by its shortest distance path

in hazard spreading graph. And $s_u = 0$ if u is the sensor vertex at the original hazard spot. Furthermore, under Assumption 6.6.3, a delay period, denoted by d_u, is considered in Eqation (2.1)for an edge to become unavailable starting from the time when the hazard is detected or arrives.

Different from the traditional adjacency matrix which uses zero for an entry corresponding to a non-existing edge, an unavailable edge is represented by an infinite number in $A^i(t)$ and A^H. Examples of the modified weighted adjacency matrices for hazard spreading and evacuation without considering congestion effect and hazard occurrence in Fig. 5.2 are expressed as

$$A^H = \begin{bmatrix} 0 & 120 & \infty & \infty & \infty & 80 \\ 120 & 0 & 120 & \infty & \infty & 110 \\ \infty & 120 & 0 & 80 & \infty & 100 \\ \infty & \infty & 80 & 0 & 120 & 110 \\ \infty & \infty & \infty & 120 & 0 & 80 \\ 80 & 110 & 100 & 110 & 80 & 0 \end{bmatrix}.$$

and

$$A^i = \begin{bmatrix} 0 & \infty & \infty & \infty & \infty & 10 \\ \infty & 0 & \infty & \infty & \infty & 15 \\ \infty & \infty & 0 & 2 & \infty & 12 \\ \infty & \infty & 2 & 0 & \infty & 15 \\ \infty & \infty & \infty & \infty & 0 & 10 \\ 10 & 15 & 12 & 15 & 10 & 0 \end{bmatrix}$$

5.3 Distributed Path Planning Algorithm for Building Evacuation under Hazard

5.3.1 Layout of the Cyber-Physical System

The CPS equipped with sensing (solid circles), information transmission (solid rectangles), and computation (white rectangles with group symbols) components for generating evacuation paths is illustrated in Fig. 5.3. Each evacuee has a smart phone or personal digital assistant (PDA) that can communicate with neighboring evacuees within certain distance to form a evacuation group, locate position, and retrieve hazard spreading information from all sensors. As stated in Assumption 5.2.1, the hazard detecting sensors represented as vertices (solid circles) are responsible for detecting the potential hazards. Once a hazard is detected, the spreading time, s_u, and delay time, d_u, will be determined based on the pre-stored hazard graph. These information will be shared by all evacuee groups to update corresponding edge cost according to (2.1) for each group. A computation node (white rectangle) is carried by one PDA selected from members of each evacuation group. It will construct evacuation graph with associated edge cost and search for evacuation paths for the his/her group only. The number of evacuees in the relative group is then forwarded to the communication nodes (solid rectangles) along corridors and around exits that will be visited by this group. The coordination protocol proposed below to prevent congestion will be performed between the computation node of each group and communication nodes to be visited.

Groups in the system will plan individual paths in a distributed manner. Each computation node is moving along the group and will plan paths for locally formed group only. After coordination, the final determined paths will be shared with all group members through personal phones or PDAs. Compared with a fixed and centralized computational framework, the mobile and distributed one prevents damages caused by hazards to a fixed computation node. Each group solves a decomposed subproblem in reduced scale, which

improves computational performance. Furthermore, it does not require collecting all evacuees' information at a centralized computation node. As long as segments (corridors and exits) from the planned paths are available, information transmission is required only between communication nodes of these segments and computation node of the corresponding evacuee group.

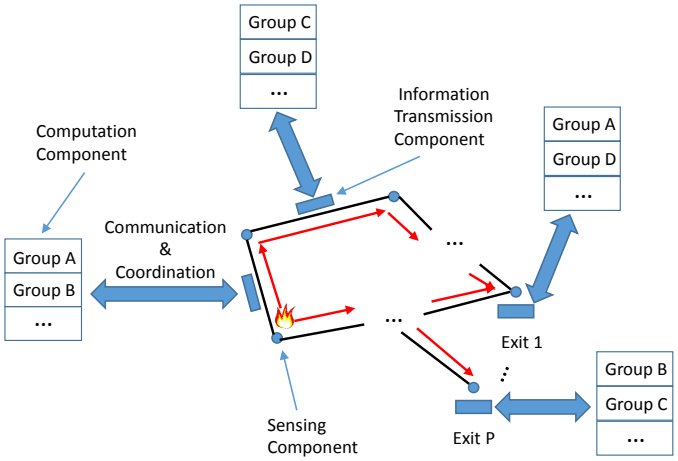

Figure 5.3 Components and information flow in CPS

5.3.2 Bellman-Ford Algorithm

In this section, Bellman-Ford algorithm [Bellman (1958)] is introduced to find the minimum time paths for each evacuee group traveling from the starting vertex to one of the exits based on the constructed evacuation graph. Bellman-Ford algorithm is an iterative approach, where current solution is replaced by a less cost solution, to find the shortest path from a source vertex to an ending vertex in a graph with weighted edges. This process is repeated until finishing examining all available paths to each ending vertex. The main procedure is illustrated in Protocol 1.

Bellman-Ford algorithm is applied in both hazard graph to generate the hazard spreading paths and evacuation graph to search for the optimal evacuation paths. For each evacuee

Define distance evaluation function $distance(k)$ and $predecessor(k)$ which measures the distance from starting node to current node k and stores predecessor of current node k, $k = 1, ..., N$.

Initialization at Time t

for *each node* $k \in 1$ *...* N **do**
  if k *is starting vertex* **then**
    $distance(k) = 0$
  end
  else
    $distance(k) = \infty$
    $predecessor(k) = null$
  end
end

Relax Edges Repeatedly

for *each node* k *excluding for the exiting node* **do**
  for *each edge* $(u, v) \in \mathcal{G}$ **do**
    if $distance(u) + A^i_{u,v}(t) < distance(v)$ **then**
      $distance(v) = A^i_{u,v}(t) + distance(u)$
      $predecessor(v) = u$
    end
  end
end

Check for Negative-Weight Cycles

for *each edge* $(u, v) \in \mathcal{G}$ **do**
  if $distance(u) + A^i_{u,v}(t) < distance(v)$ **then**
    error: graph contains negative-weight cycle.
  end
end

Protocol 5 Bellman-Ford Algorithm Applied to Finding Shortest Evacuation Path

group, Bellman-Ford algorithm searches for the minimum cost paths from the starting vertex representing evacuee group's current location to the ending vertices, i.e. available exits. After evaluating cost of available edges connecting to each exit, the optimal path with minimum cost is selected. Bellman-Ford algorithm simply relies on topology and edge cost of a specific weighted graph and thus evacuation path for each group can be determined separately. An example based on the evacuation graph of Fig. 5.1 is illustrated in Fig. 5.4. A evacuee group starting from the room of left bottom corner is guided to the exit through a minimum time path labeled in red color.

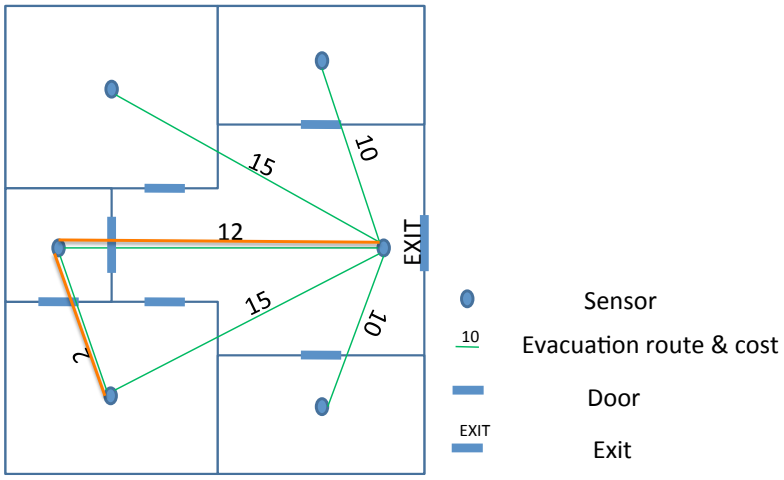

Figure 5.4 Example of one evacuation route computed from Bellman-Ford method.

5.3.3 Problem Formulation and Decomposition

A distributed coordination protocol is developed to avoid congestion and thus improve evacuation efficiency. In the distributed path planning approach, the entire evacuation planning problem is decomposed into a set of subproblems with associated objective function and constraints. Each subproblem can be solved individually by the corresponding computation node using the Bellman-Ford algorithm introduced above. The individual solutions

are then coordinated through the subgradient method to satisfy the specified constraints. The integrated Bellman-Ford and dual subgradient algorithm, named as BFDS is described in this section.

The path planning problem for minimum time evacuation with capacity constraints along corridors and around exits are expressed as

$$J = \min_x \Sigma_{i=1}^{M} f^i(x^i) \tag{3.4}$$
$$s.t. \quad g_j(x,t) \leq 0,\ j = 1,\ldots,P,\ t = 1,\ldots,T$$
$$l_{u,v}(x,t) \leq 0,\ (u,v) \in \mathcal{G},\ t = 1,\ldots,T,$$

where x^i, $i = 1,\ldots,M$, is the designated edges in the evacuation graph to guide evacuees of group i to the exit, the objective function $f^i(x^i)$ represents time consumption for group i reaching available exit, $g_j(x,t) \leq 0$ is the capacity constraint at exit j for time interval t, $l_{u,v}(x,t) \leq 0$ is the capacity constraint for the edge (u,v), T is the number of time intervals, and there are P exits in the building.

The capacity constraints for exits are formulated as

$$g_j(x,t) = n_j(t) - n_j^{max} \leq 0,\ j = 1,\ldots,P,\ t = 1,\ldots,T,$$

where $n_j(t)$ is the number of evacuees in vicinity of exit j at time interval t, n_j^{max} is the capacity of exit j, which represents the maximum number of evacuees allowed in vicinity of exit j if it is available, and $n_j(t)$ at time interval t is obtained via

$$n_j(t) = \begin{cases} n_j(t-1) + (\bar{\delta}^T(t)\bar{r} - V_j)\tau, \ if\ n_j(t-1) + \bar{\delta}_t(T)\bar{r}\tau > V_j\tau \\ 0 \qquad\qquad otherwise, \end{cases} \tag{3.5}$$

where V_j is the egress ability of exit j, i.e. the maximum number of evacuees allowed to pass through exit j per unit time τ, $\bar{r} = [r_1,\ldots,r_M]^T$ is the set of flow rate for all evacuee groups, representing the number of evacuees passing through a defined space per unit time [Wang et al. (2009)], $\bar{\delta}(t) = [\delta_1(t),\ldots,\delta_M(t)]^T$ and each element in $\bar{\delta}(t)$ is determined by

$$\delta_i(t) = \begin{cases} 1, \text{ if group } i \text{ arrives exit } j \text{ at time interval } t \\ 0, \quad \text{otherwise.} \end{cases}$$

For exit j, $\bar{n}_j = [n_j(1), \ldots, n_j(T)]^T$ and each element of $\bar{n}_j$ is determined via (3.5). If group i is going to visit exit j, it will send $\delta_i(t)$ with associated r_i to communication node at exit j. By receiving information from all groups that will visit exit j, $\bar{n}_j$ will be determined. The capacity constraints for edges representing corridors of a building are formulated as

$$l_{u,v}(x,t) = n_{u,v}(t) - n_{u,v}^{max} \leq 0,\ (u,v) \in \mathcal{G},\ t = 1, \ldots, T,$$

where $n_{u,v}^{max}$ is the capacity of edge (u,v), $n_{u,v}(t)$ is the total number of evacuees passing through edge (u,v) at time interval t which can be obtained via

$$n_{u,v}(t) = n_{u,v}(t-1) + \bar{\rho}^T(t)\bar{r}\tau \tag{3.6}$$

where $\bar{\rho}(t) = [\rho_1(t), \ldots, \rho_M(t)]^T$ and each element in $\bar{\rho}(t)$ is determined by

$$\rho_i(t) = \begin{cases} 1, & \text{if head of group } i \text{ is in } (u,v) \text{ and tail is out of } (u,v) \text{ at time interval } t \\ -1, & \text{if tail of group } i \text{ is in } (u,v) \text{ and head is out of } (u,v) \text{ at time interval } t \\ 0, & \text{otherwise.} \end{cases}$$

Similar to finding $n_j(t)$, by collecting information through communication node along edge (u,v) from all groups that visit this edge at time interval t, $n_{u,v}(t)$ will be determined.

From the above definitions, the Lagrangian for problem (3.4) is expressed as,

$$L = \Sigma_{i=1}^{M} f^i(x^i) + \Sigma_{j=1}^{P}(\bar{\lambda}_j)^T(\bar{n}_j - n_j^{max}) + \Sigma_{(u,v)\in\mathcal{G}}(\bar{\mu}_{u,v})^T(\bar{n}_{u,v} - n_{u,v}^{max}), \tag{3.7}$$

where $\bar{\lambda}_j = [\lambda_j(1), \ldots, \lambda_j(T)]^T$ and $\bar{\mu}_{u,v} = [\mu_{u,v}(1), ..., \mu_{u,v}(T)]^T$ are Lagrangian multipliers, $\bar{n}_{u,v} = [n_{u,v}(1), ..., n_{u,v}(T)]^T$ and $\bar{n}_j = [n_j(1), ..., n_j(T)]^T$. Under Assumption 6.6.2, it is assumed that evacuees in the same room or neighborhood will follow others' movement. According to localization and vicinity of scattered evacuees, for M locally formed evacuee groups, the above optimization problem with constraints can be decomposed into M sub-problems. As a result, the evacuation path for each group is calculated separately through independent computation nodes. The ith Lagrangian is formulated as

$$L^i = f^i(x^i) + \Sigma_{t=1}^{T}\lambda_j(t)n_{last,j}^i(t) + \Sigma_{(u,v)\in\mathcal{G}_i}\Sigma_{t=1}^{T}\mu_{u,v}(t)n_{u,v}^i(t), \tag{3.8}$$

where $n^i_{last,j}(t)$ denotes the number of evacuees from group i along the last edge connecting to exit j, $n^i_{u,v}(t)$ denotes the number of evacuees from group i passing through edge (u,v) at time interval t, and $\mathcal{G}_i$ indicates the evacuation graph for group i. At each iteration, solution of x^i can be determined from Bellman-Ford algorithm to minimize the "phantom" cost by adding $\Sigma_{t=1}^{T}\lambda_j(t)n^i_{last,j}(t) + \Sigma_{(u,v)\in\mathcal{G}_i}\Sigma_{t=1}^{T}\mu_{u,v}(t)n^i_{u,v}(t)$ to the original cost function, $f^i(x^i)$. The Lagrangian multipliers associated with time interval t are updated via

$$\lambda_j^{q+1}(t) = \max[0, \lambda_j^q(t) + \alpha_g^q g_j(x^q(t))] \tag{3.9}$$

$$\mu_{u,v}^{q+1}(t) = \max[0, \mu_{u,v}^q(t) + \alpha_l^q l_{u,v}(x^q(t))]. \tag{3.10}$$

By utilizing the CPS described in Section 5.3.1, each group will construct the corresponding evacuation graph by adding "phantom" cost on edges. For example, the additional cost $\lambda_j(t)n^i_{last,j}(t)$ will be added to the last edge segment connecting to exit j. The iterative coordination procedure starts by assuming $\bar{\lambda}_j = \mathbf{0}$ for $j = 1,\ldots,P$ and $\bar{\mu}_{u,v} = \mathbf{0}$ for $(u,v) \in \mathcal{G}$ in the first step, $q = 1$. After each group finding their evacuation path, x^i, in the first step, the communication nodes along corridors and around exits will receive information, i.e., δ_i, r_i, and ρ_i, from visiting groups. After receiving these information, the Lagrangian multipliers will be updated through the communication nodes according to (3.9) and (3.10) and then sent back to relative groups to update evacuation graph for the next iteration. The local path planning for individual groups and coordination through communication nodes is repeated until $|\bar{\lambda}_j^{q+1} - \bar{\lambda}_j^q| \leq \varepsilon_\lambda$ and $|\bar{\mu}_{u,v}^{q+1} - \bar{\mu}_{u,v}^q| \leq \varepsilon_\mu$, where ε_λ and ε_μ are thresholds for stopping criteria. The proposed BFDS algorithm is summarized as in Protocol 6.

5.4 Simulation and Discussion

This section provides two simulation examples, a simple graph with a few number of vertices and edges and a real world building with complicated floor plans. without loss of generality, we assume that each group possesses identical moving capability. Hence the

superscript i in $f^i_{u,v}[n(t)]$, $F^i_{u,v}[n(t)]$, and $c^i_{u,v}$ are ignored and we simply use $f_{u,v}$, $F_{u,v}$, and $c_{u,v}$, respectively.

5.4.1 Case 1: A Simple Graph

In the first case, simulation is carried out based on a simple graph which abstractly represents a object building including 8 vertices and 11 edges. The corresponding evacuation model is developed based on a 8×8 weighed adjacency matrix with off-diagonal entries assigned by integer numbers between $[1, 10]$ if an edge exists. We assign all vertices from one to eight and two exits are labeled as #5 and #8. Each vertex, excluding the two exits, has one evacuee group starting from there. The number of evacuees in each group is randomly selected from the range of $[1, 10]$. The edge cost functions, $f_{u,v}$ and $F_{u,v}$, are assumed to be the same. They are monotonically increasing with increment of the number of evacuees and expressed as

$$f_{u,v} = F_{u,v} = c_{u,v} + 2(n_{u,v}(t) - 7),\ if\ n_{u,v}(t) \geq 7. \tag{4.11}$$

The above linear function indicates that the edge cost increases 2 seconds when every one additional evacuee is involved along path segment (u, v) if the group has more than 7 evacuees. Otherwise, the edge cost is equal to $c_{u,v}$. Relevant parameters are specified in Table 5.1.

Table 5.1 Parameter settings for case 1

Exit Capacity $[n_5^{max}, n_8^{max}]$	$[2, 2]$ $person$
Egress Ability $[V_5, V_8]$	$[1, 1]$ $person/s$
Edge Capacity $n_{u,v}^{max}$	15 $person$
Unit Time τ	$1s$
Simulation Time T	$100s$

The BFDS algorithm is implemented in the simple case without considering spreading of

hazards. To demonstrate the improved performance of the proposed method, the shortest path solution is provided as well. The comparative solution does not consider cooperation among evacuee groups. Thus, each group find their shortest route individually. Figures 5.5 and 5.6 demonstrate the number of evacuees in vicinity of each exit during the evacuation process. The comparative results indicate that the peak number at exit 1 is reduced from 15 to 2 using the BFDS algorithm and the exit capacity constraints are satisfied. The shortest path always guide evacuee groups to the nearest exit without considering congestion along path segments or near exits. While result from BFDS reduces the overall evacuation time from 34 to 27 seconds. Figure 5.7 illustrates the successful avoidance of path congestion and jams at exits through cooperation between group 1 and 2. The planned route from BFDS algorithm for each evacuee group is represented by dash colored lines in Fig. 5.7 and the alternative route based on shortest path is represented by corresponding solid colored lines.

In order to verify performance of the proposed BFDS algorithm, we also solve Case 1 in a centralized manner with evacuation graph information shared among all evacuation groups. The evacuation path generated from the centralized algorithm yields the same result as the one from BFDS algorithm, shown in Fig. 5.7. However, the centralized algorithm considering hazard and capacity constraints depends on reliable communication channels and high performance computation node to collect all required information and then generate evacuation paths simultaneously. Any communication disconnection and sensor malfunction will lead to a failure.

5.4.2 Case 2: Real Application in Building Evacuation

In the real world case, simulation is carried out based on the floor plans of Howe Hall at Iowa State University. As illustrated in Fig. 5.10, this building includes faculty offices, class rooms, and graduate student workplaces. There are 79 sensors scattered on this floor and each room has one sensor for hazard detecting. The rest of those sensors are installed

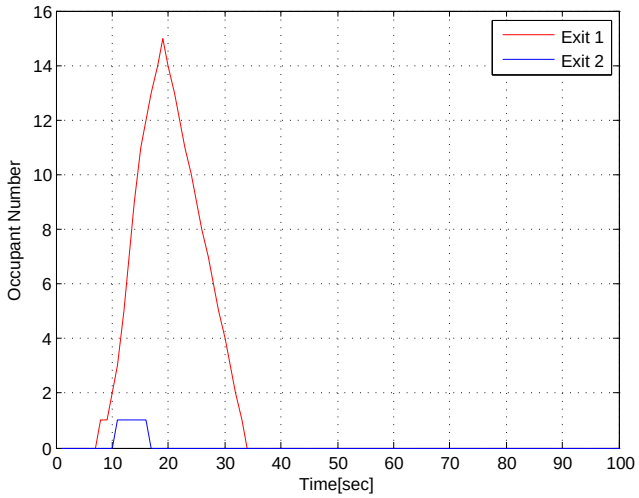

Figure 5.5 Case 1: Number of evacuees at exit 1 (red) and exit 2 (blue) using shortest path

in corridors and close to exits. Blue colored segments are used to denote possible egress paths between rooms, corridors, and exits in Fig. 5.10. By assigning edge cost obtained through virtual computer simulation and experimental investigation, evacuation graph is developed and represented by a 79×79 weighted adjacency matrix, $\mathcal{A}$. Moreover, three exits are labeled as #20, #58, and #79.

To avoid congestion and reduce "faster is slower" effect, strict capacity constraints for corridors and vicinity area of exits are considered. For the object building in the simulation case, the parameters related to the exit capacity constraints are set as, $n_{20}^{max} = 6$, $n_{58}^{max} = 5$ and $n_{79}^{max} = 7$. All relevant simulation parameters are listed in Table. 5.2. The number of evacuees in each group is randomly generated from the range of $[a, b]$. Distribution of $[a, b]$ for different room sizes is illustrated in Table. 5.3.

To simulate the dynamic hazard influence on evacuation path planning, we assume the carpet is ignited close to sensor #5. After detecting the heat and smoke, sensor #5 reports the hazard location to the guidance system and predicts the fire spreading paths along every possible edges to obtain t_u for each vertex using Bellman-Ford algorithm. When updating the edge cost of the evacuation graph at each step, the corresponding edge cost is set as

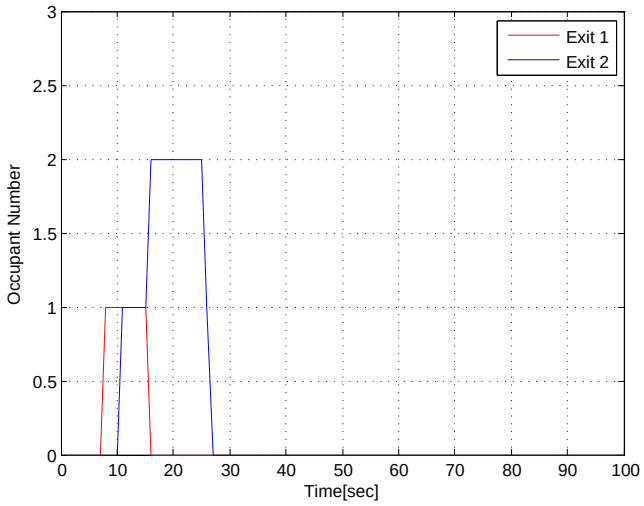

Figure 5.6 Case 1: Number of evacuees at exit 1 (red) and exit 2 (blue) using BFDS algorithm

an infinite number based on (2.1) if the last person in a particular group cannot reach the sensor vertex before hazard arrives.

Again, solution from shortest path is provided to compare with results obtained from the proposed BFDS algorithm. Figures 5.8 and 5.9 demonstrate time history of evacuee numbers in vicinity of each exit from both methods. It is shown that the proposed BFDS algorithm significantly reduces the number of evacuees in order to satisfy capacity constraints at exit. Furthermore, Fig. 5.10 illustrates changing of a typical planned path that successfully avoid hazard, where the original paths are represented by red solid lines and the changed paths are represented by red dash lines. The proposed BFDS algorithm also alleviates path congestion demonstrated by overlapped routes in green, yellow, and purple colors. Finally, the overall evacuation time from the shortest path is 184 seconds while the one from BFDS is 64 seconds which is less than half of the time required by the shortest path solution.

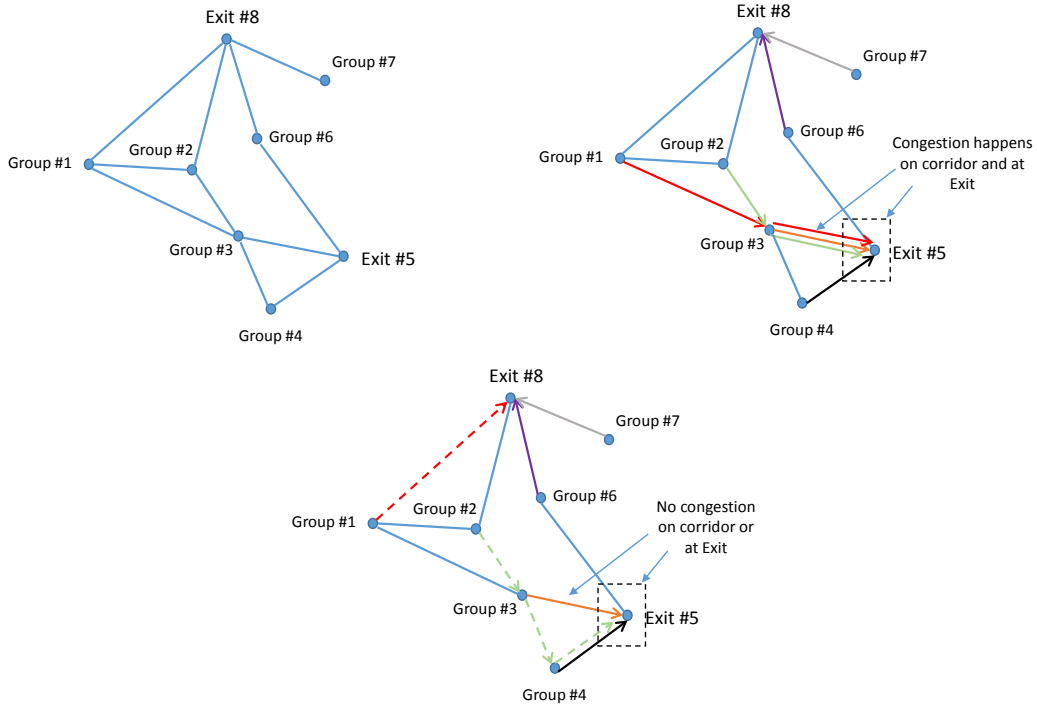

Figure 5.7 *Upper Left*: evacuation graph in Case 1. *Upper Right*: evacuation paths by shortest path algorithm (Solid Arrows). *Lower*: changed paths (dash arrows) and unchanged paths (solid arrows) by centralized and BFDS algorithms considering congestion constraints.

5.5 Conclusion

We propose a time-efficient evacuation guidance strategy utilizing a cyber-physical system. The approach first constructs a graph based dynamic evacuation model with precisely estimated edge costs considering hazard spreading and congestion effects. Based on the dynamic graph, we convert the evacuation path planning problem to a network optimization problem subject to capacity constraints. By incorporating dual subgradient method and Bellman-Ford algorithm, a Bellman-Ford-Dual-Subgradient (BFDS) algorithm is proposed to find minimum time evacuation paths in a distributed framework. Compared to the shortest path algorithm where each group finds corresponding shortest evacuation route individually, the BFDS algorithm coordinates planned paths among groups via sensing and

Table 5.2 Parameter settings for case 2

Exit Capacity $[n_{20}^{max}, n_{58}^{max}, n_{79}^{max}]$	$[6, 5, 7]$ $person$
Egress Ability $[V_{20}, V_{58}, V_{79}]$	$[2, 2, 4]$ $person/s$
Edge Capacity $n_{u,v}^{max}$	15 $person$
Unit Time τ	$1s$
Simulation Time T	$300s$
Path Unavailable Time Delay d_u	$5s$
Hazard Starting Position	#5

Table 5.3 Occupants distribution

Room Classification	Distribution Range $[a, b]$
Large Class Room	$[0, 30]$
Small Class room	$[0, 15]$
Large graduate student workspace	$[0, 20]$
Small graduate student workspace	$[0, 10]$
Windows Computer Lab	$[0, 30]$
Faculty Office	$[0, 2]$
Near Sensor in Corridor	$[0, 2]$

communication networks. Simulation examples verify that the integrated dynamic evacuation model and BFDS algorithm efficiently find routes for all evacuee groups while satisfying strict exit capacity constraints. Meanwhile, the proposed method alleviates congestion occurred along corridors and around exits during the evacuation. More importantly, the overall evacuation time will be significantly reduced, especially for cases with large number of evacuees and limited available exits.

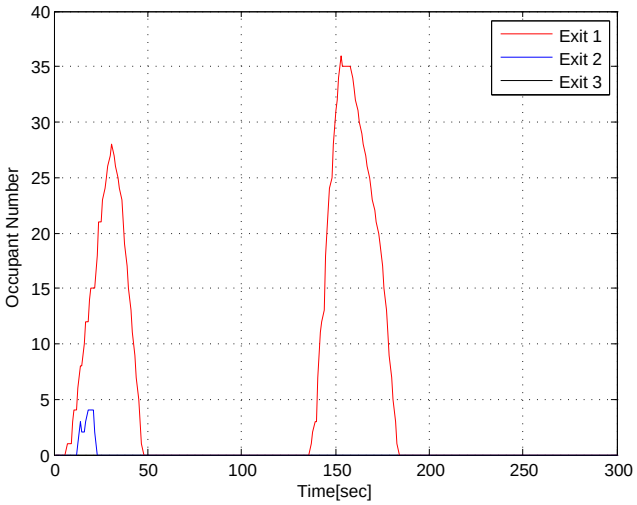

Figure 5.8 Case 2: Occupant number obstructed at exit 1 (red), exit 2 (blue) and exit 3 (black) using shortest path finding

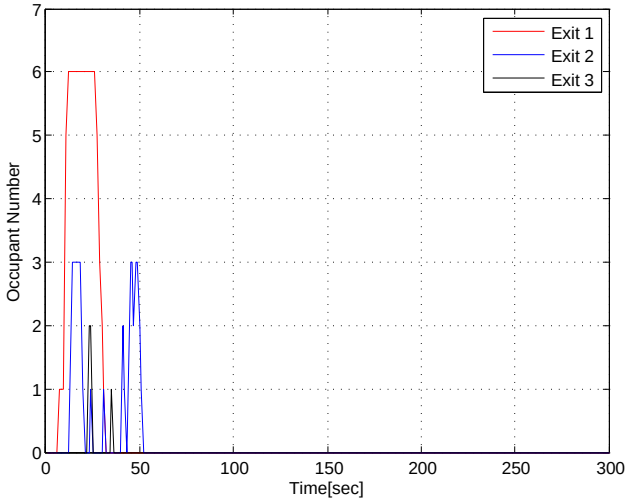

Figure 5.9 Case 2: Occupant number obstructed at exit 1 (red), exit 2 (blue) and exit 3 (black) using BFDS algorithm

Initialization: $q = 1$, $\bar{\lambda}^1 = \mathbf{0}$, $\bar{\lambda}^2 = \inf$, $\bar{\mu}^1 = \mathbf{0}$, $\bar{\mu}^2 = \inf$;
for *Group* $i \leftarrow 1$ **to** M **do**
    Use Bellman-Ford method to find minimum cost path for each group by solving Eq. (3.8).
end
while $|\bar{\lambda}_j^{q+1} - \bar{\lambda}_j^q| \geq \varepsilon_\lambda$ *or* $|\bar{\mu}_{u,v}^{q+1} - \bar{\mu}_{u,v}^q| \geq \varepsilon_\mu$, *for* $j = 1, \ldots, P$ *and* $(u, v) \in \mathcal{G}$ **do**
    $q = q + 1$
    Update evacuation model for each group based on Eq. (2.1).
    for $t \leftarrow 1$ **to** T **do**
        Calculate $n_j(t)$ via Eq. (3.5).
        $g_j(x, t) = n_j(t) - n_j^{max}$.
        Calculate $n_{u,v}(t)$ via Eq. (3.6).
        $l_{u,v}(x, t) = n_{u,v}(t) - n_{u,v}^{max}$.
    end
    for *Exit* $j \leftarrow 1$ **to** P **do**
        Update $\bar{\lambda}_j^q$ via Eq. (3.9).
        for *Group* $i \in N_j$ *where* N_j *represents the set of groups visiting exit* j **do**
            $A_{last,j}^i = A_{last,j}^i + \sum_{t=1}^T \lambda_j^q(t) n_{last,j}^i(t)$, where $A_{last,j}^i$ corresponds to the cost for the last edge connecting to exit j in the evacuation graph of group i.
        end
    end
    for *Edge* $(u, v) \in \mathcal{G}$ **do**
        Update $\bar{\mu}_{u,v}^q$ via Eq. (3.10).
        for *Group* $i \in N_{u,v}$ *where* $N_{u,v}$ *represents the set of groups going through edge* (u, v) **do**
            $A_{u,v}^i = A_{u,v}^i + \sum_{t=1}^T \mu_{u,v}^q(t) n_{u,v}^i(t)$.
        end
    end
    Use Bellman-Ford algorithm to update shortest path for each group.
end

Protocol 6 BFDS in-building path planning algorithm

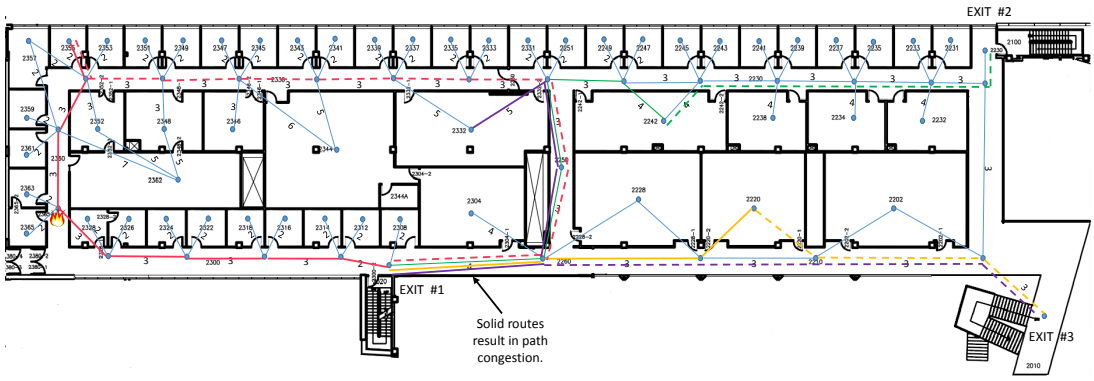

Figure 5.10 Case 2: Evacuation paths from shortest path finding (solid line)and BFDS (dash line)

CHAPTER 6. DISTRIBUTED OPTIMIZER DESIGN: INTELLIGENT HIGHWAY TRAFFIC MANAGEMENT

6.1 Problem Statement

A one-dimensional, uniform highway section considered in this project is represented by $[\xi, \chi]$, where ξ and χ are upstream and downstream boundaries. We denote the vehicle density as $\rho(t, x)$ per unit length for local position $x \in [\xi, \chi]$ at time $t \in [0, t_M]$. The inflow and outflow are denoted as Q_ξ and Q_χ, respectively. The vehicle velocity is a function of ρ and is denoted as $v = v(\rho(t, x))$. The goal of the proposed traffic control strategy is to minimize the fuel consumption, or total travel time of vehicles on the concerned highway section for a desired time interval based on current traffic status by controlling dynamic speed limit signals, shown as an example in Fig. 6.1, where the arrows represent on-ramps and off-ramps.

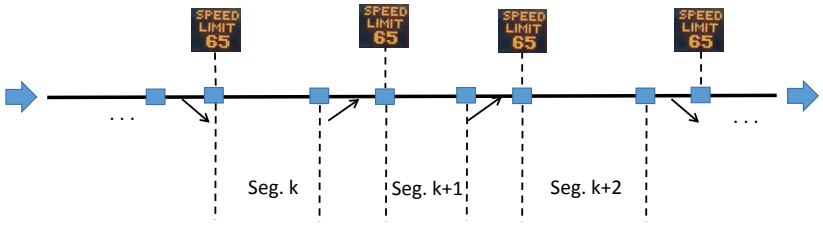

Figure 6.1 Example of a traffic control scenario. Arrows at upstream and downstream boundaries refer to the vehicle flow directions. Arrows along main road section represent on-ramps and off-ramps. Rectangles indicate installed sensors for measuring traffic volume. Dynamic speed limit signs are located at the starting point of each road segment.

6.2 Traffic Flow Dynamics: LWR Model and Moskowitz Function

The macroscopic traffic flow model was first introduced by Ligthill and Whitham in the 1950s [Lighthill and Whitham (1955)] and was intensively investigated afterwards. The fundamental traffic flow model is based on the continuous conservation law in the form of partial differential equations (PDE). For the one-dimension, uniform highway section considered above, the total number of vehicles conserved in time interval $[t, t + \Delta t]$ can be expressed as

$$\int_{\xi}^{\chi} (\rho(t + \Delta t, x) - \rho(t, x))dx = (Q_{\xi} - Q_{\chi})\Delta t. \tag{2.1}$$

Intuitively, the integral of the changing density with respect to local position x is the number of conserved vehicle during a small time interval Δt. Assuming inflow Q_{ξ} and outflow Q_{χ} are constant during the small time interval Δt, dividing both sides of the above expression by Δt gives

$$\int_{\xi}^{\chi} \frac{\partial \rho}{\partial t} dx = \rho(t, \xi)v(\rho(t, \xi)) - \rho(t, \chi)v(\rho(t, \chi)). \tag{2.2}$$

The right hand side of the above equation is based on the traffic flow theory, $Q = \rho v$, where v is the velocity and it only depends on the density such that $v = v(\rho(t, x))$. Further simplifying the above equation follows

$$\int_{\xi}^{\chi} \frac{\partial \rho}{\partial t} dx = -\int_{\xi}^{\chi} \frac{\partial (\rho v)}{\partial x} dx. \tag{2.3}$$

Since equality in (2.3) must be satisfied for all t and x, combining the integrand of both sides leads to the well-known LWR PDE written as

$$\frac{\partial \rho}{\partial t} + \frac{\partial Q(\rho(t, x))}{\partial x} = 0. \tag{2.4}$$

The LWR PDE is the fundamental traffic flow model based on the continuous conservation law. Now introducing the cumulated vehicle count $N(t,x)$, the vehicle density and flow can be calculated directly from the partial derivatives with respect to local position x and time t in forms of

$$\rho(t,x) = -\frac{\partial N(t,x)}{\partial x} \tag{2.5}$$

$$Q(t,x) = \frac{\partial N(t,x)}{\partial t}. \tag{2.6}$$

Substituting $\rho(t,x)$ and $Q(t,x)$ in (2.4) by (2.5) and (2.6) and then integrating both sides with respect to the local position generates Moskowitz HJ PDE,

$$\frac{\partial N(t,x)}{\partial t} - Q(-\frac{\partial N(t,x)}{\partial x}) = 0. \tag{2.7}$$

Considering the initial, upstream, and downstream boundary conditions, i.e. $c_{ini}(x)$, $c_{up}(t)$ and $c_{down}(t)$, together with the Moskowitz HJ PDE, the Cauchy problem [Mazaré et al. (2011)] is formulated as,

$$\begin{cases} (2.7) \\ N(0,x) = c_{ini}(x) \\ N(t,\xi) = c_{up}(t) \\ N(t,\chi) = c_{down}(t). \end{cases} \tag{2.8}$$

The traffic control optimization problem is to minimize the value of a specific objective, e.g. the fuel consumption, or total travel time, while satisfying the four equality constraints of the Cauchy problem listed above by designing control variables $v(t,x)$.

6.3 B/J-F Explicit solutions to Traffic Flow Dynamics

Inspired by B-J/F solution for HJ PDEs [Barron and Jensen (1990)], we adopt B-J/F solution to Moskowitz HJ PDEs [Lighthill and Whitham (1955); Richards (1956); Bayen et al. (2007)] to obtain the solutions for estimating the traffic flow states in any future time.

The solution can be explicitly expressed based on a fundamental diagram [Greenshields et al. (1935)] associated with initial and boundary conditions. Furthermore, the solution to Moskowitz HJ PDEs can be simplified based on roadway decomposition and traffic status.

6.3.1 Piecewise Affine Initial and Boundary Conditions

We discretize time period $[0, t_M]$ and highway section $[\xi, \chi]$ into several small intervals using time step T and spatial step X. The initial vehicle density, $\rho(0, x_k)$, $k = 0, 1, 2..., k_m$, is assumed to be identical within segment $[x_k, x_{k+1}]$. Inflow and outflow remain constant during each time interval $[t_n, t_{n+1}]$ indexed by $n = 0, 1, ..., n_m$. For the time step T, it is constant during the entire controlling period and cannot be chosen as a large value due to the assumption of constant density during each time interval. However, T has to be greater than the lower bound, $\frac{X}{v_f}$, which is the minimum time for the inflow traffic arriving the downstream segment of each divided highway segment. By picking such lower bound, it is possible to observe the inflow traffic at the downstream segment. Another reason of avoiding small T value is that control information is required to transmit for displaying and the traffic must have enough time to adjust their speed to obey the newly updated speed limit. Time step T could be time-varying in the future in accordance with the inflow traffic measurement and prediction. For the spatial step, the notation X remains consistent for different highway segments in this section. However, it can be different in real-world scenario and we have specified the spatial step based on different length of segments for explicit solution derivation. Highway section is segmented at every location where on-ramp or off-ramp appears. Due to the assumption of identical density within a highway segment, each length should be small, e.g. generally hundreds of meters. If that is still a long distance after spatial decomposition based on ramp location, we further divide such segment.

The initial and boundary conditions, c_{ini}, c_{up}, and c_{down}, can be decomposed into affine, locally-defined condition set, i.e. c^k_{ini}, c^n_{up}, and c^n_{down}. For example, the negative initial condition, $-c^k_{ini}(t, x)$, represents the total number of vehicles at initial time contained between

$[\xi, x]$. The upstream condition, $c^n_{up}(t,x)$, and downstream condition, $c^n_{down}(t,x)$, depicts total number of vehicles entering and exiting the roadway from initial to current time t. Hence we summarize the piecewise affine equations regarding initial and boundary conditions as [Canepa and Claudel (2012)],

$$c^k_{ini}(t,x) = \begin{cases} -\sum_{i=0}^{k-1} \rho(0,x_i)X - \rho(0,x_k)(x-x_k), \text{ if } t=0 \ \& \ x \in [x_k, x_{k+1}] \\ +\infty, \quad otherwise \end{cases} \tag{3.9}$$

$$c^n_{up}(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q(t_i,\xi)T + Q(t_n,\xi)(t-nT), \text{ if } x=\xi \ \& \ t \in [t_n, t_{n+1}] \\ +\infty, \quad otherwise \end{cases} \tag{3.10}$$

$$c^n_{down}(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q(t_i,\chi)T + Q(t_n,\chi)(t-nT) - \sum_{k=0}^{k_m} \rho(0,x_k)X, \\ \quad \text{if } x=\chi \,\&\, t \in [t_n, t_{n+1}] \\ +\infty. \quad otherwise \end{cases} \tag{3.11}$$

Based on semi-explicit espressions of solution to Moskowitz HJ PDE, presented in Claudel and Bayen (2010a), we introduce triangular-model-based solutions in Appendix A. Moreover, Greenshields-model-based B-J/F explicit solution are developed and obtained associated with initial and boundary conditions. Derivation details and result expressions can be found in Appendix B. In the following sections, Greenshields-model-based B-J/F explicit solutions are simplified by considering two assumptions.

6.3.2 General Semi-Explicit Solution

We aggregate initial, upstream, and downstream boundary conditions in a value condition function, $\mathbf{c}(t,x)$. From the work in Aubin et al. (2008); Claudel and Bayen (2010a,b) the solutions $N_{\mathbf{c}}$ associated by value condition function $\mathbf{c}$ is the infimum of infinite number of value condition functions. Then the B-J/F solution to (2.7) can be represented by

$$N_{\mathbf{c}}(t,x) = \inf_{(u,T)\in[w,v_f]\times\mathbb{R}_+} [\mathbf{c}(t-T, x-Tu) + TR(u)]. \tag{3.12}$$

By using the fundamental diagram which define a flow function of traffic density $Q(\rho)$, convex transform $R(u)$ is defined as follows

$$R(u) = \sup_{\rho\in[0,\rho_j]} (Q(\rho) - u\rho),\ \forall u \in [w, v_f], \tag{3.13}$$

with $w = \frac{dQ}{d\rho}|_{\rho=\rho_j} < 0$, $\rho_j > 0$, and $v_f = \frac{dQ}{d\rho}|_{\rho=0} > 0$ denoting the jam density and free-flow speed, respectively.

6.3.3 Simplified B-J/F Explicit Solution

Due to the piecewise affine property, triangular-model-based B-J/F solution can be directly incorporated into the model constraints which is introduced in III.F. However, Greenshields model results in non-determined piecewise nonlinear B-J/F solutions. In order to construct linear model constraints based on Greenshields-based solutions, we simplify (2.8-2.11) in this section. First two assumptions are required for solution simplification.

Assumption 6.3.1. *A one lane highway with long distance can be decomposed into several segments with distance X^k. B-J/F solution can be implemented in each segment.*

Assumption 6.3.2. *The highway section is required to handle cases with relatively large vehicle flow, i.e. flow at origin and ending is close to the road capacity Q_c. In other word, one has $(1 - \frac{1}{\sqrt{q}})\rho_c^k \le \rho_{up}^k \le \rho_c^k$ and $\rho_c^k \le \rho_{down}^k \le (1 + \frac{1}{\sqrt{q}})\rho_c^k$, where q is a user-specified parameter determining bounds of constraints. Substituting the above bounds on ρ_{up}^k and ρ_{down}^k in $T_0(\rho_{up}^k)$ and $T_0(\rho_{down}^k)$ expressions, respectively, yields $T_0(\rho_{up}^k) \ge \sqrt{q}\frac{x-\xi^k}{v_f^k}$ and $T_0(\rho_{down}^k) \ge \sqrt{q}\frac{\chi^k - x}{v_f^k}$.*

kth road segment $[\xi^k, \chi^k]$ is regarded as an individual object with associated length X^k, jam density ρ_j^k, and free-flow speed v_f^k after decomposition. Assumption 6.3.1 simply sets the initial density to be $\{\rho(0, 0^k), \rho(0, X^k)\}$ and denotes $\rho_{ini}^k = \rho(0, 0^k)$ as vehicle density for each segment. 0^k denotes the starting point of segment k. Furthermore, the plot of function (2.10) in Fig. 6.2 demonstrates that the slope of tangent line at each time instance increases

when t varies from $t_n + \frac{x-\xi^k}{v_f^k}$ to $\frac{v_f^k \rho_j^k}{4}$. Similar conclusion can be derived from the solution curve associated with the downstream boundary condition. Assumption 6.3.2 introduces a linear approximation for (2.10) when $t \geq t_n + T_0(\rho_{up}^k)$ and (2.11) when $t \geq t_n + T_0(\rho_{down}^k)$. Based on these discussion, the initial and boundary conditions for each road segment with modified notation $Q_{up}^{(t,x_k)} = Q(t, 0^k)$, $Q_{down}^{(t,x_k)} = Q(t, X^k)$, and ρ_{ini}^k, is expressed as

$$c_{ini}^0(t,x) = \begin{cases} -\rho_{ini}^k x, & \text{if } t = 0 \ \& \ x \in [0^k, X^k] \\ +\infty \quad otherwise \end{cases} \tag{3.14}$$

$$c_{up}^n(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q_{up}^{(t_i,x_k)} T + Q_{up}^{(t_n,x_k)}(t - nT), \ \text{if } x = 0^k \ \& \ t \in [t_n, t_{n+1}] \\ +\infty \quad otherwise \end{cases} \tag{3.15}$$

$$c_{down}^n(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q_{down}^{(t_i,x_k)} T + Q_{down}^{(t_n,x_k)}(t - nT) - \rho_{ini}^k X^k, \ \text{if } x = X^k \ \& \ t \in [t_n, t_{n+1}] \\ +\infty \quad otherwise. \end{cases} \tag{3.16}$$

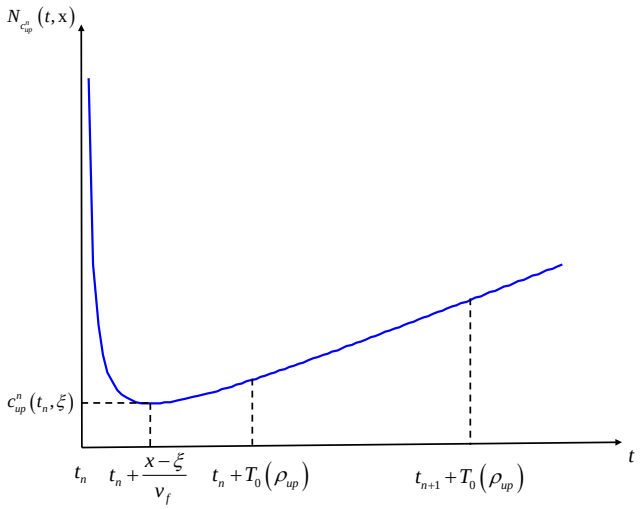

Figure 6.2 Sketch of function (2.10).

Since initial value condition is reduced to only one segment $[0^k, X^k]$, number of vehicles on $[0^k, X^k]$ at initial time can be calculated by $\rho_{ini}^k X^k$ from (3.14) and (3.16). With the up-

dated initial and boundary conditions, the B-J/F solution associated with initial conditions in (3.14)-(3.16) is simplified in (2.8)-(2.11) by setting $\xi = 0^k$ and $\chi = X^k$. Furthermore, from Assumption 6.3.2, we use the formal definition of a linear function as the simplified solution when time is greater than the corresponding threshold. For initially free-flow case with $0 \leq \rho_{ini}^k \leq \rho_c^k$, it reduces to

$$N_{c_{ini}^0}(t,x) = \begin{cases} (-\frac{v_f^k}{\rho_j^k}(\rho_{ini}^k)^2 + v_f\rho_{ini})t - \rho_{ini}^k x, \text{ if } 0 \leq t \leq \dfrac{x}{Q'(\rho_{ini}^k)} \\ \dfrac{v_f^k}{4}\rho_j^k t + \dfrac{x^2\rho_j^k}{4v_f^k t} - \dfrac{x}{2}\rho_j^k, \text{ if } t \geq \dfrac{x}{Q'(\rho_{ini}^k)} \end{cases} \tag{3.17}$$

and for initially congested case with $\rho_c^k \leq \rho_{ini}^k \leq \rho_j^k$, it becomes

$$N_{c_{ini}^0}(t,x) = \begin{cases} (-\frac{v_f^k}{\rho_j^k}(\rho_{ini}^k)^2 + v_f^k\rho_{ini}^k)t - \rho_{ini}^k x, \text{ if } 0 \leq t \leq \dfrac{x-X}{Q'(\rho_{ini}^k)} \\ \dfrac{v_f^k}{4}\rho_j^k t + \dfrac{(x-X^k)^2\rho_j^k}{4v_f^k t} - \dfrac{x-X^k}{2}\rho_j^k - \rho_{ini}^k X^k, \text{ if } t \geq \dfrac{x-X^k}{Q'(\rho_{ini}^k)}. \end{cases} \tag{3.18}$$

The solution components associated with boundary conditions become

$$N_{c_{up}^n}(t,x) = \begin{cases} \dfrac{(v_f^k - \frac{x}{t-t_n})^2\rho_j^k}{4v_f^k}(t-t_n) + \sum_{i=0}^{n-1} Q_{up}^{(t_i,x_k)}T, \text{ if } t_n \leq t \leq t_n + \sqrt{q}\dfrac{x}{v_f^k} \\ a(t-t_n) + b, \quad \text{if } t \geq t_n + \sqrt{q}\dfrac{x}{v_f^k} \end{cases} \tag{3.19}$$

$$N_{c_{down}^n}(t,x) = \begin{cases} \dfrac{(v_f^k - \frac{X^k-x}{t_n-t})^2\rho_j^k}{4v_f^k}(t-t_n) + \sum_{i=0}^{n-1} Q_{down}^{(t_i,x_k)}T - \rho_{ini}^k X^k, \\ \qquad \text{if } t_n \leq t \leq t_n + \sqrt{q}\dfrac{X^k-x}{v_f^k} \\ e(t-t_n) + f, \text{ if } t \geq t_n + \sqrt{q}\dfrac{X^k-x}{v_f^k} \end{cases} \tag{3.20}$$

where a, e are slopes of the tangent line at corresponding time and b, f are the relative function values at $t = t_n + \sqrt{q}\frac{x}{v_f^k}$ and $t = t_n + \sqrt{q}\frac{X^k-x}{v_f^k}$.

6.3.4 Model Constraints

To make the B-J/F solutions compatible with value conditions, a infinit number of equality constraints must be hold in Cauchy problem 2.8. Based on the Inf-morphism

property [Claudel (2010)] and Lax-Hopf formula in (3.12), the last three equalities in the Cauchy problem can be converted into a set of inequalities.

Lemma 6.3.1. *Compatibility Conditions, [Li et al. (2014)]: The solution to HJ PDE is characterized by the Inf-morphism property, i.e.* $\boldsymbol{c}(t,x) = \min_{l\in L} c_l(t,x)$*, where* L *is the index number of value condition, the solution* $N_{\boldsymbol{c}}(t,x) = \min_{l\in L} N_{c_l}(t,x)$ *for* $(t,x) \in [0,t_M]\times[\xi,\chi]$*. The B-J/F solution to (2.7) satisfies the value conditions if and only if*

$$N_{c_i}(t,x) \geq c_j(t,x),\ \forall (t,x) \in Dom(c_j),\ (i,j) \in L^2. \tag{3.21}$$

Inequalities in (3.21) represent the model constraints. By considering these constraints, the B-J/F solutions are reduced to a subset representing the exact solution to the Cauchy problem. Solution to HJ PDE can be explicitly expressed based on Lax-Hopf formula. We integrate these expressions with piecewise affine value conditions to formulate model constraints

As described in §III, B-J/F solution is the exact solution to Cauchy problem if inequality of (3.21) holds. We reduce these continuous inequalities for $\forall (t,x) \in Dom(c_j)$ into a series of discrete inequalities by discretizating the continuous time interval into a set of small time intervals with step size $T = 1\,sec$. By utilizing the linear interpolation on $[pT,(p+1)T)$, the piecewise affine functions are built with respect to time t. Therefore, the discrete inequality constraints are expressed in (3.22).

$$\begin{cases}
(i)\ N_{c^n_{up}}(0,x) \geq c^k_{ini}(0,x) \ \ \forall (n,k) \in \{0,...,n_m\}\times\{0,...,k_m\}\ \&\ x \in [x_k, x_{k+1}] \\
(ii)\ N_{c^k_{ini}}(t,\xi^k) \geq c^p_{up}(t,\xi^k) \ \ \forall (k,p) \in \{0,...,k_m\}\times\{0,...,n_m\}\ \&\ t \in [pT,(p+1)T] \\
(iii)\ N_{c^n_{down}}(0,x) \geq c^k_{ini}(0,x) \ \ \forall (n,k) \in \{0,...,n_m\}\times\{0,...,k_m\}\ \&\ x \in [x_k, x_{k+1}] \\
(iv)\ N_{c^k_{ini}}(t,\chi^k) \geq c^p_{down}(t,\chi^k) \ \ \forall (k,p) \in \{0,...,k_m\}\times\{0,...,n_m\}\ \&\ t \in [pT,(p+1)T] \\
(v)\ N_{c^n_{up}}(t,\chi^k) \geq c^p_{down}(t,\chi^k) \ \ \forall (n,p) \in \{0,...,n_m\}^2\ \&\ t \in [pT,(p+1)T] \\
(vi)\ N_{c^n_{down}}(t,\xi^k) \geq c^p_{up}(t,\xi^k) \ \ \forall (n,p) \in \{0,...,n_m\}^2\ \&\ t \in [pT,(p+1)T]
\end{cases} \tag{3.22}$$

Constraints (i) and (iii) in (3.22) are satisfied for $x \in [\xi^k, \chi^k]$, $t \in [0, t_M]$ in the simplified solution [Claudel and Bayen (2010a)]. The remaining constraints in (3.22) are replaced by corresponding expressions defined in (3.14)-(3.20). For initially free-flow condition with $\rho_{ini} \leq \rho_c$ and discrete time index $p \in [n, n_m]$ for $t \in [pT, (p+1)T]$, constraints (ii) and (iv) in (3.22) become

$$(ii) \quad \frac{v_f^k \rho_j^k}{4} t \geq Q_{up}^{(t_p,x_k)}(t - pT) + \sum_{l=0}^{p-1} Q_{up}^{(t_l,x_k)} T \tag{3.23}$$

$$(iv) \begin{cases} (-\frac{v_f^k}{\rho_j^k}(\rho_{ini}^k)^2 + v_f^k \rho_{ini}^k)t \geq Q_{down}^{(t_p,x_k)}(t - pT) + \sum_{l=0}^{p-1} Q_{down}^{(t_l,x_k)} T, \text{ if } 0 \leq t \leq \dfrac{X^k}{Q'(\rho_{ini}^k)} \\ (\dfrac{v_f^k \rho_j^k}{4} - Q_{down}^{(t_p,x_k)})t^2 + (Q_{down}^{(t_p,x_k)} pT - \sum_{l=0}^{p-1} Q_{down}^{(t_l,x_k)} T + (\rho_{ini}^k - \dfrac{\rho_j^k}{2})X^k)t \\ + \dfrac{(X^k)^2 \rho_j^k}{4 v_f^k} \geq 0, \text{ if } t \geq \dfrac{X^k}{Q'(\rho_{ini}^k)}. \end{cases} \tag{3.24}$$

For initially congested conditions with $\rho_{ini}^k \geq \rho_c^k$, constraints (ii) and (iv) in (3.22) become

$$(ii) \begin{cases} (-\frac{v_f^k}{\rho_j^k}(\rho_{ini}^k)^2 + v_f^k \rho_{ini}^k)t \geq Q_{up}^{(t_p,x_k)}(t - pT) + \sum_{l=0}^{p-1} Q_{up}^{(t_l,x_k)} T \quad \text{if } 0 \leq t \leq \dfrac{-X}{Q'(\rho_{ini}^k)} \\ (\dfrac{v_f^k \rho_j^k}{4} - Q_{up}^{(t_p,x_k)})t^2 + (Q_{up}^{(t_p,x_k)} pT - \sum_{l=0}^{p-1} Q_{up}^{(t_l,x_k)} T + (\rho_{ini}^k - \dfrac{\rho_j^k}{2})X^k)t \\ + \dfrac{(X^k)^2 \rho_j^k}{4 v_f^k} \geq 0, \text{ if } t \geq \dfrac{-X^k}{Q'(\rho_{ini}^k)}, \end{cases} \tag{3.25}$$

$$(iv) \quad \frac{v_f^k \rho_j^k}{4} t \geq Q_{down}^{(t_p,x_k)}(t - pT) + \sum_{l=0}^{p-1} Q_{down}^{(t_l,x_k)} T. \tag{3.26}$$

For $t_n \leq t \leq t_n + \sqrt{q}\frac{X^k}{v_f^k}$, constraints (v) and (vi) in (3.22) become

$$\begin{aligned} (v)\, (\frac{v_f^k \rho_j^k}{4} - Q_{down}^{(t_p,x_k)})(t - nT)^2 + (W^k + Q_{down}^{(t_p,x_k)}(p - n)T \\ - \sum_{l=n}^{p-1} Q_{down}^{(t_l,x_k)} T)(t - nT) + \frac{(X^k)^2 \rho_j^k}{4 v_f^k} \geq 0 \end{aligned} \tag{3.27}$$

$$\begin{aligned} (vi)\, (\frac{v_f^k \rho_j^k}{4} - Q_{up}^{(t_p,x_k)})(t - nT)^2 + (-W^k + Q_{up}^{(t_p,x_k)}(p - n)T \\ - \sum_{l=n}^{p-1} Q_{up}^{(t_l,x_k)} T)(t - nT) + \frac{(X^k)^2 \rho_j^k}{4 v_f^k} \geq 0, \end{aligned} \tag{3.28}$$

where $W^k = \sum_{l=0}^{n-1}(Q_{up}^{(t_l,x_k)} - Q_{down}^{(t_l,x_k)})T + (\rho_{ini}^k - \frac{\rho_j^k}{2})X^k$. For $t \geq t_n + \sqrt{q}\frac{X^k}{v_f^k}$, constraints (v) and (vi) in (3.22) become

$$(v)\,(a - Q_{down}^{(t_p,x_k)})(t - nT) + Q_{down}^{(t_p,x_k)}(p-n)T + \rho_{ini}^k X^k + b - \sum_{l=0}^{p-1} Q_{down}^{t_l} T \geq 0 \tag{3.29}$$

$$(vi)\,(e - Q_{up}^{(t_p,x_k)})(t - nT) + Q_{up}^{(t_p,x_k)}(p-n)T + f - \sum_{l=0}^{p-1} Q_{up}^{(t_l,x_k)} T \geq 0. \tag{3.30}$$

Equations (3.23)-(3.30) are model constraints describing traffic flow dynamics. Constraints (ii) in (3.23) and (iv) in (3.26) have been verified since $\frac{v_f^k \rho_j^k}{4} \geq \max_{t\in[0,t_M]}\{Q_{up}^{(t,x_k)}, Q_{down}^{(t,x_k)}\}$. Hence both of them are ignored in formulation of the following optimization problem.

6.4 Real-Time Energy-Efficient Traffic Control via Convex Optimization

In the first application of developed model constraints based on Greenshield fundamental diagram, we build a quadratic cost function in terms of vehicle volume to estimate fuel consumption rate based on COPERT model. Benefit from the affinity of model constraints, a convex quadratic optimization problem is then formulated to generate energy-efficient traffic control decisions in real-time. Simulation results demonstrate significant reduction of fuel consumption on testing highway sections under peak traffic demands of busy hours.

6.4.1 A General Formulation of Fuel Efficiency Transportation Control Problem

The COPERT model is a macroscopic model estimating the emission and fuel consumption rate based on average vehicle speed [Zegeye (2011)]. The quadratic form of emission or fuel consumption objective with respect to average speed for different vehicle classes, such as v_{gp} and v_{dp}, is expressed as

$$J = w_{gp}(c_{gp0}v_{gp}^2 + c_{gp1}v_{gp} + c_{gp2}) + w_{dp}(c_{dp0}v_{dp}^2 + c_{dp1}v_{dp} + c_{dp2}) + ..., \tag{4.31}$$

where the quadratic parameters are specified in terms of vehicle category, for example, quadratic coefficients, denoted by c_{gp} are for gasoline passenger cars, c_{dp} for diesel passenger cars, and etc. The weighting factors, such as w_{gp} and w_{dp}, determined from sensor measurements, are proportional to number of vehicle counted from different classes. In this work, additional inflow and outflow contributed from on-ramp and off-ramp are considered. The volume on each off-ramp is assumed to be proportional to corresponding main highway section volume with a constant ratio $R_{off}^{x_k}$. And on-ramp vehicle volume is a constant, denoted by $C_{on}^{x_k}$. Thus additional linear equality constraints related to inflow and outflow are included in the problem formulation. Considering both ramp-effect constraints and the linear model constraints in terms of $Q_{up}^{(t_n,x_k)}$ and $Q_{down}^{(t_n,x_k)}$, the fuel efficient traffic control problem is formulated as

$$
\begin{aligned}
min.\ \ & J = (4.31) \\
s.t.\ \ & A_{model}\mathbf{y} \leq b_{model} \\
& Q_{down}^{(t_n,x_k)} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if no ramp exists on } [x_k, x_{k+1}], \\
& (1 - R_{off}^{x_k}) Q_{down}^{(t_n,x_k)} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if off-ramp exists on } [x_k, x_{k+1}], \\
& Q_{down}^{(t_n,x_k)} + C_{in}^{x_k} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if on-ramp exists on } [x_k, x_{k+1}],
\end{aligned}
\tag{4.32}
$$

where A_{model} and b_{model} represent the parameter matrix and vector derived from the linear model constraints. The unknown variable set, $\mathbf{y} = [Q_{down}^{(t_n,x_0)},\ Q_{up}^{(t_n,x_0)},$ ..., $Q_{down}^{(t_n,x_{k_m})},\ Q_{up}^{(t_n,x_{k_m})}]^T$, which includes inflow and outflow at time instant t_n for all segments. By solving the above problem, we find the optimized inflow and outflow variables for each segment during $[t_n, t_{n+1}]$. From the determined $Q_{up}^{(t_n,x_k)}$ and $Q_{down}^{(t_n,x_k)}$ for $k = 0, 1, ..., k_m$, the desired vehicle density for next time interval can be obtained from

$$
\rho(t_{n+1}, x_k) = \frac{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + \rho(t_n, x_k)X^k}{X^k} \tag{4.33}
$$

based on the conservation law. To reach the desired vehicle density at next time instant t_{n+1}, the desired speed of each segment, denoted by $v_d(t_n, x_k)$, at time interval $[t_n, t_{n+1}]$ is determined by

$$v_d(t_n, x_k) = -\frac{v_f^k}{\rho_j^k}\rho(t_{n+1}, x_k) + v_f^k. \tag{4.34}$$

From (4.34), optimized inflow and outflow decision variables are converted into desired speed for segment k during $[t_n, t_{n+1}]$ which are the control variables and are expressed as,

$$\mathbf{v} = [v_d(t_n, x_0),\ v_d(t_n, x_1),\ ...,\ v_d(t_n, x_{k_m})]^T. \tag{4.35}$$

6.4.2 Triangular-Model-Based Problem Formulation

To simplify the demonstration of different objective functions in the following sections, we assume all vehicles belong to class of EURO I or onwards, the speed range is $13.1 - 130\,km/h$ and for each vehicle the cylinder capacity range is $1.41L - 2.01L$. Although the formulation from the triangular fundamental diagram cannot lead to a convex problem that guarantees real-time optimal solution, the triangular model in (1.1) has been recognized as a more accurate model when representing the flow-density relationship. Thus the off-line solution from the triangular model provides a reference to evaluate effectiveness of the CQOP formulation based on the Greenshields Model. The estimates of the COPERT fuel consumption in (4.31) using the triangular model is formulated as

$$\begin{aligned}
J^{tri} &= \sum_{k=0}^{k_m} X^k[c_0 v_a(t_{n+1}, x_k)^2 + c_1 v_a(t_{n+1}, x_k) + c_2] \\
&= \sum_{k=0}^{k_m} X^k[c_0\omega^2(1 - \frac{\rho_j^k}{\rho_a(t_{n+1}, x_k)})^2 + c_1\omega(1 - \frac{\rho_j^k}{\rho_a(t_{n+1}, x_k)}) + c_2] \\
&= \sum_{k=0}^{k_m} X^k[c_0\omega^2(1 - \frac{X^k\rho_j}{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + X^k\rho(t_n, x_k)})^2 \\
&\qquad + c_1\omega(1 - \frac{X^k\rho_j}{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + X^k\rho(t_n, x_k)}) + c_2], \\
&\qquad \text{if } \rho_c \le \rho_a \le \rho_j.
\end{aligned} \tag{4.36}$$

The above cost function denotes the fuel consumed on road section $[\xi, \chi]$ during time interval $[t_{n+1}, t_{n+2}]$ given density $\rho(t_n, x_k)$. In (4.36), we define the average vehicle speed at time t and location x as $v_a(t,x) \in [0, v_f]$. Hence the second equality holds due to $v_a = \omega(1 - \frac{\rho_j}{\rho})$. Moreover, we further express the average density at next time interval as $\rho_a(t_{n+1}, x_k) = \frac{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + \rho(t_n,x_k)X^k}{X^k}$, which leads to the third equality in (4.36). Since the average velocity is the free-flow velocity when $0 \leq \rho_a < \rho_c$, we ignore the free-flow case. Using the triangular model, the general formulation for the fuel efficiency transportation problem in (4.32) is expressed as

$$\begin{aligned} min.\ \ & J^{tri} = (4.36) \\ s.t.\ \ & A_{model}^{tri}\mathbf{y} \leq b_{model}^{tri} \\ & ramp\ constraints\ in\ (4.32), \end{aligned} \tag{4.37}$$

where A_{model}^{tri} and b_{model}^{tri} are the parameter matrix and vector of linear constraints derived from the triangular model. However, the objective formulated in (4.36) is nonlinear which requires a Nonlinear Programming (NLP) solver to solve the above problem without guarantee of convergence. For real-time traffic control, the nonlinear formulation and existing NLP solvers are not reliable.

6.4.3 Greenshields-Model-Based Problem Formulation

From the Greenshields fundamental diagram, one has $v_a = -\frac{v_f}{\rho_j}\rho_a(t,x) + v_f$. Then the performance index is constructed as

$$\begin{aligned} J &= \sum_{k=0}^{k_m} X^k [c_0 v_a(t_{n+1}, x_k)^2 + c_1 v_a(t_{n+1}, x_k) + c_2] \\ &= \sum_{k=0}^{k_m} X^k [c_0 (-\frac{v_f^k}{\rho_j^k}\rho_a(t_{n+1}, x_k) + v_f^k)^2 + c_1 (-\frac{v_f^k}{\rho_j^k}\rho_a(t_{n+1}, x_k) + v_f^k) + c_2] \\ &= \sum_{k=0}^{k_m} X^k [c_0 (-\frac{v_f^k}{\rho_j^k}\frac{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + \rho(t_n,x_k)X^k}{X^k} + v_f^k)^2 \\ &\qquad + c_1 (-\frac{v_f^k}{\rho_j^k}\frac{(Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})T + \rho(t_n,x_k)X^k}{X^k} + v_f^k) + c_2] \end{aligned} \tag{4.38}$$

The quadratic form of the objective function is determined by $Q_{up}^{(t_n,x_k)}$ and $Q_{down}^{(t_n,x_k)}$. Moreover, the Hessian matrix of the above objective function in (4.38) is expressed as

$$H = \begin{bmatrix} p^0 & -p^0 & 0 & 0 & ... & 0 \\ -p^0 & p^0 & 0 & 0 & ... & 0 \\ 0 & 0 & p^1 & -p^1 & ... & 0 \\ 0 & 0 & -p^1 & p^1 & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & ... & 0 & 0 & p^{k_m} & -p^{k_m} \\ 0 & ... & 0 & 0 & -p^{k_m} & p^{k_m} \end{bmatrix}$$

where $p^k = 2c_0(\frac{v_f^k}{\rho_j^k}\frac{T}{X^k})^2$. Since the Hessian matrix derived above is positive semidefinite, it implies the cost function in (4.38) is a convex function. The corresponding problem formulation based on the Greenshields diagram is expressed as

$$\begin{aligned} min.\ \ & J^{Gre} = (4.38) \\ s.t.\ \ & A_{model}^{Gre}\mathbf{y} \leq b_{model}^{Gre} \\ & ramp\ constraints\ in\ (4.32), \end{aligned} \tag{4.39}$$

which is a CQOP, where A_{model}^{Gre} and b_{model}^{Gre} represent the parameter matrix and vector form for model constraints (3.23)-(3.30). Due to the convexity, a global optimum for (4.39) could be obtained within polynomial computational time using the existing convex optimization solver [Grant and Boyd (2008)].

6.4.4 Simulation and Discussion

6.4.4.1 Real-World Scenario and VISSIM Setup

The test highway section is a $7.42km$ long spatial domain of I-235 from 50th Street to exit 5B, which is one of the busiest freeways in West Des Moines, Iowa. The existing Iowa Department of Transportation (Iowa DOT) Wavetronix sensors, which are used to capture

traffic data, cover the highway network of West Des Moines and Des Moines. The collected aggregated data was obtained through an online data portal maintained by TransCore. The weekday data for morning peak (7:00 a.m. to 9:00 a.m.) from May 1st to September 30th, 2015 is used in this work. Based on Greenshields model, linear regression is used to fit the speed-density line illustrated in Fig. 6.3 where one example fitting line for the highway segment from Valley West Dr. (NB) to exit 2 is shown. Parameters related to fundamental diagram can be calculated graphically. By making speed equal to zero, jam density ρ_j^k can be derived accordingly. Similarly, the free flow speed v_f^k is obtained when density is zero.

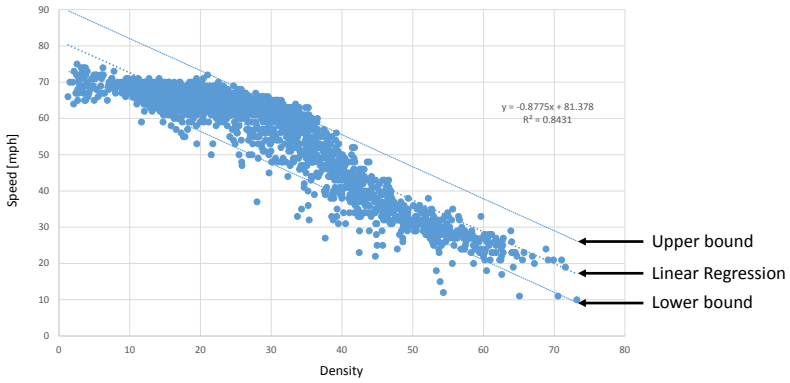

Figure 6.3 Speed-Density line fitted through linear regression

The proposed traffic control strategy by solving the formulated NLP or CQOP is originally programmed in Matlab. To build the connection of control program with VISSIM simulation, we generate a COM interface. The COM interface module is designed to access all network object attributes and realize the user-defined control algorithms [VISION (2014)]. Through the COM interface, most of the simulation parameters can be dynamically handled through programming [Shou-feng et al. (2012); Tettamanti and Varga (2012)].

According to the ramp location, test highway section is divided into 10 segments. As an example, segment 4 is shown in Fig. 6.4. There are four sensors installed at the starting point of segment 4, where each of them records volume entering into segment 4 on corresponding lanes. Another four volume sensors are set at the ending point of segment 4,

collecting the traffic volume flowing out. The volume records return to zero for every 120 seconds. Dynamic speed limit signs are located in accordance with the physical characteristics of each highway segment. Location of the speed limit sign for the first segment is at the starting point of the test freeway. Locations of the rest speed limit signs could be right after an on-ramp or an off-ramp. For example, in Fig. 6.4, the speed limit sign is located at the starting point of segment 4, which is right after the on-ramp of I-235 EB at Valley West Dr (NB). The highway segment description is shown in Fig. 6.1. Based on the method proposed by Shaw and Noyce [Shaw and Noyce (2014)], the traffic volume of study corridor can be balanced. We offer the raw observation volume in Fig. 6.1 as well. Furthermore, ramp length is not considered in the simulation scenarios so that there is no ramp length value provided in Fig. 6.1.

VISSIM has two car following models, Wiedemann 74 for urban traffic, and Wiedemann 99 for freeway traffic. The Wiedemann 99 car following model is used in this study. Driver behavior parameters are calibrated before simulation. Three parameters, standstill distance (CC0), headway time (CC1), and 'following' variation (CC2), are found to have significant influences on traffic capacity in calibration. The calibrated CC0 is $3.05m$, CC1 is $1.45s$, and CC2 is $7.41m$. More details about the calibration is described in Dong et al. (2015).

In the following simulation scenarios, real-time control to minimize fuel consumption is achieved by the following procedures. First, traffic volume of each segment in time interval $[t_{n-1}, t_n]$ is collected by volume sensors installed before and after each entry or exit point where the vehicles are guided into or leaving the main highway section. Second, desired vehicle density at t_n is obtained through (4.33) which is assumed to be constant during $[t_n, t_{n+1}]$. Third, based on current density information at t_n, we solve NLP or CQOP so that optimized inflow and outflow can be determined. Fourth, desired density for t_{n+1} is calculated through (4.33). At the last step, we attain desired speed during $[t_n, t_{n+1}]$ by (4.34) and return $v_d(t_n, x_k)$ as corresponding dynamic speed limit sign. The time interval for speed limit updating is 120 seconds.

Figure 6.5 illustrates the entire procedures of determining the desired speed during $[t_n, t_{n+1}]$. To verify improvement of fuel efficiency, the fuel consumption amount with and without the control strategy is recorded and compared. The default speed limit of the test section is 120 km/h for the case without speed control. For each scenario, the simulation is designed to last 4200 seconds. Since traffic status is not stable at the beginning period, only the simulation results from 600 to 4200 seconds are used for data analysis. To demonstrate feasibility of the proposed control strategy, we consider four scenarios under different volume demands, including the original traffic flow on I-235 and inflow from I-35N and 50th Street N. For scenarios 1-4, the volume demands are specified as, 4500, 5000, 5500, and 6000 veh/h, respectively. Simulation results are discussed in the following section.

Figure 6.4 The Test Segment of I-235 from Valley West Dr (NB) to exit 2 in Des Moines, IA. Arrows indicate location of volume sensors. Elliptical regions imply the entry or exit location for on-ramp or off-ramp vehicles. Dynamic speed limit sign is at the left vertex of the segment.

6.4.4.2 Comparison of Two Types of Formulation

In this section, results obtained from the NLP formulation in (4.37) and CQOP in (4.39) are presented and discussed. Considering the aforementioned test highway section of I-235, a quadratic programming (QP) solver is used to solve the CQOP in (4.39). Two types of NLP solvers, interior-point and sequential quadratic programming (SQP), are used to

Segment Description				Raw Observed Data			
Order	Station	Type	Length (km)	Volume (vph)			
1	I-235 EB @ 50nd St (NB)	Starting point		4500	5000	5500	6000
2	Segment 1	Main	0.72	4500	5000	5500	6000
3	Exit 1B	Off-ramp		1215	1350	1485	1620
4	Segment 2	Main	0.75	3285	3650	4015	4380
5	I-235 EB @ Valley West Dr (SB)	On-ramp		452	452	452	452
6	Segment 3	Main	0.33	3737	4102	4467	4832
8	I-235 EB @ Valley West Dr (NB)	On-ramp		660	660	660	660
7	Segment 4	Main	0.50	4397	4762	5127	5492
9	Exit 2	Off-ramp		2111	2286	2461	2633
10	Segment 5	Main	0.99	2286	2476	2666	2859
11	I-235 EB@ 22nd St	On-ramp		744	744	744	744
12	Segment 6	Main	0.66	3030	3220	3410	3603
13	Exit 3	Off-ramp		939	1000	1057	1117
14	Segment 7	Main	0.55	2091	2222	2353	2486
15	I-235 EB @ 8th St Loop	On-ramp		408	408	408	408
16	Segment 8	Main	0.38	2499	2603	2761	2894
17	Exit 4	Off-ramp		350	341	386	405
18	Segment 9	Main	1.07	2149	2262	2375	2489
19	I-235 EB @ 63rd St	On-ramp		260	260	260	260
20	Segment 10	Main	1.47	2409	2522	2635	2749
21	I-235 EB @ 42nd St	Off-ramp		554	580	606	632

Table 6.1 Test highway description and balanced observation data

solve the NLP problem formulated in (4.37). Parameters in both QP and NLP solvers are set to be the same, including function tolerance (10^{-6}), constraints tolerance (10^{-6}), and maximum iteration number (4000). Table I compares the performance of two NLP solvers and a QP solver when solving the corresponding formulated problems. In order to analyze dependence of initial states for NLP solvers, we randomly generate 100 groups of initial states and percentage of convergence for each NLP solver is shown in Table I. The QP solver does not require initial guess.

Table 6.2 Performance comparison of NLP solvers (interior-point and SQP) for NLP in (4.37) and a QP solver for CQOP in (4.39)

	interior-point	SQP	QP
requirement of initial guess	YES	YES	NO
optimal solution type	LOCAL	LOCAL	GLOBAL
# of iterations	189	182	11
computational time	$0.65s$	$0.51s$	$0.17s$
percentage of convergence	16%	66%	100%

From Table. 6.2, it is apparent that convergence of NLP solvers depends on appropriate selection of initial states. However, it is challenging to find appropriate initial states at each time instant to guarantee local convergence in real-time computation. Even though local

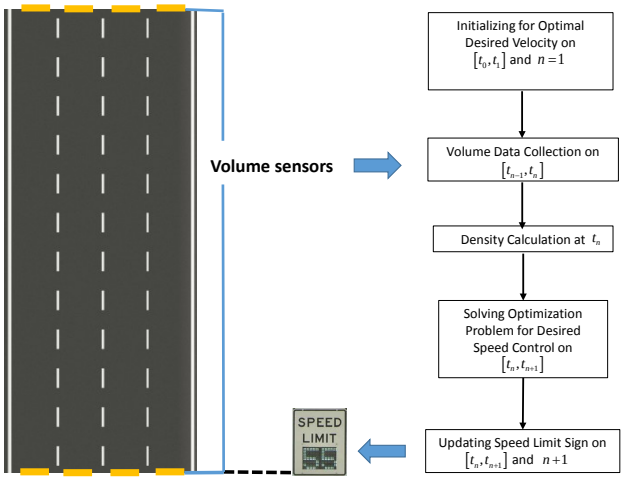

Figure 6.5 Real-Time optimal control procedures

convergence is obtained for some cases, performance of the objective value is not guaranteed. In addition, an NLP solver generally takes more time to find a solution even for converged cases. On the other side, the CQOP in (4.39) can be solved via a QP solver to obtain a global optimal solution with much less computational time.

To verify the accuracy of the CQOP formulation, results from CQOP using QP solver is compared with the convergent solution from NLP formulation using SQP solver in Fig. 6.6, where the optimal controlled speed and traffic density for every highway segments are shown for both methods. Results from both NLP and CQOP are very close, which indicates the high precision of CQOP formulation. Moreover, the fuel consumption during this time interval from both methods is shown in Table 6.3, which again demonstrates the consistency of two solutions. Based on the above comparison and discussion, in the following simulation, we choose the Greenshields-model-based CQOP formulation that significantly improves the computational efficiency without sacrificing accuracy.

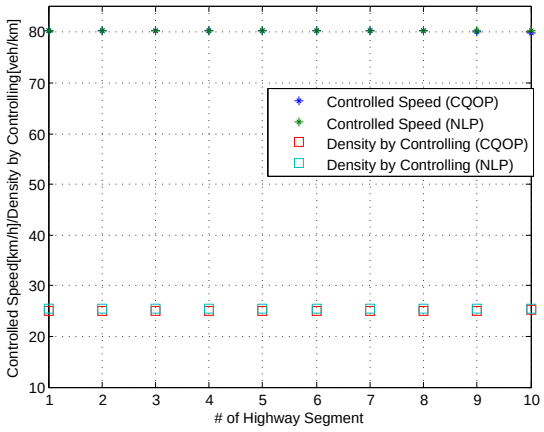

Figure 6.6 Controlled speed and traffic density by solving NLP in (4.37) and CQOP in (4.39).

Table 6.3 Comparison of fuel consumption formulated in (4.36) and (4.38)

	w/o control	with control	fuel reduction
Amount in (4.36) (g)	388.91	315.18	18.95%
Amount in (4.38) (g)	394.29	317.02	19.60%
relative difference	1.38%	0.58%	

6.4.4.3 Simulation Results

As shown in Fig. 6.3, we determine a parallelogram region by shifting fitted speed-density line up and down. The speed could be slightly different from the theoretical result provided by linear regression in a neighborhood. Hence, to consider realistic application, optimal speed value is rounded to the increment of 5 km/h and no less than 15 km/h. Figure 6.7 illustrates histories of control variables, i.e. suggested driving speed shown on speed limit signs. Figures 6.8-6.11 demonstrate the density history with and without speed control for simulation scenarios 1 through 4, respectively. Density history diagram demonstrates the average density reduction excluding the first two segments. The proposed control strategy leads to lower average vehicle density compared with uncontrolled one. Especially for segment 9 that generates a high density value at the end of simulation period,

our algorithm avoids severe congestion for that segment. Figures 6.9-6.11 demonstrate the improved performance on congestion alleviation when relatively high demanding ($\geq$ 5000 veh/h) exists. Quantitative comparison of Total Travel Time (TTT) are shown in Table . Proposed controller effectively reduces the TTT by congestion alleviation.

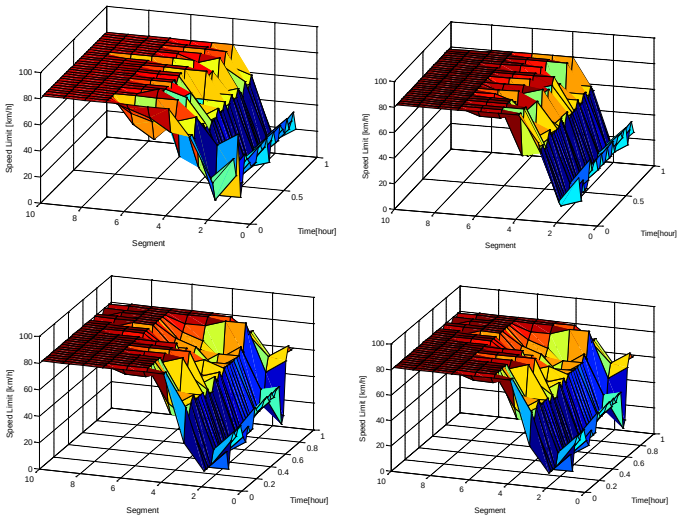

Figure 6.7 Histories of Controlled Speed Limit. *Upper Left*: 4500 veh/h Demand at Origin. *Upper Right*: 5000 veh/h Demand at Origin. *Lower Left*: 5500 veh/h Demand at Origin. *Lower Right*: 6000 veh/h Demand at Origin.

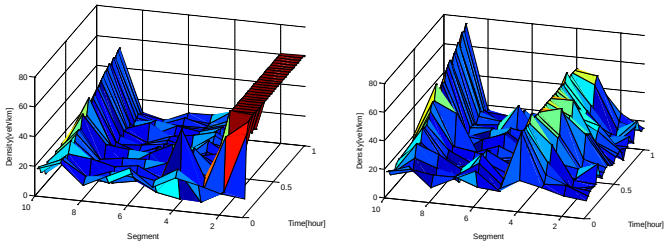

Figure 6.8 Density History of Scenario 1 for 4500 veh/h Demand at Origin. *Left*: Speed limit signs are controlled by the rounded optimal solution. *Right*: Uncontrolled case with desired speed of 120km/h for each segment.

Total fuel consumption of vehicles on test highway section during simulation time is provided in Table 6.5. We pick five different seed parameters to initialize five random generators in VISSIM. Different seed settings allow us to simulate stochastic variations of vehicles entering test freeway at the origin. Five sets of comparison results are shown in

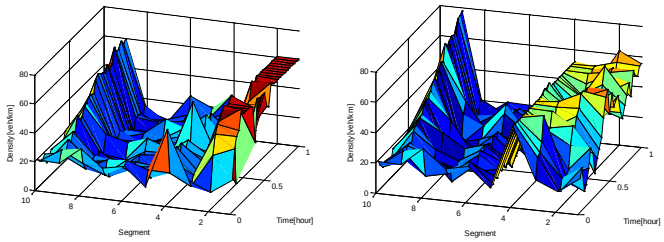

Figure 6.9 Density History of Scenario 2 for 5000 veh/h Demand at Origin. *Left*: Speed limit signs are controlled by the rounded optimal solution. *Right*: Uncontrolled case with desired speed of 120km/h for each segment.

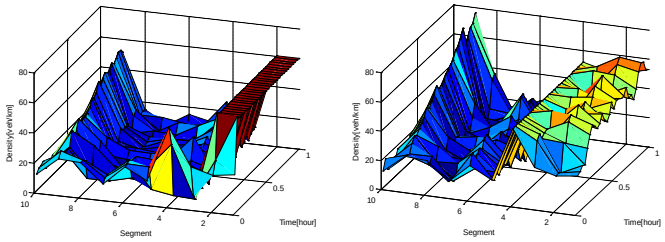

Figure 6.10 Density History of Scenario 3 for 5500 veh/h Demand at Origin. *Left*: Speed limit signs are controlled by the rounded optimal solution. *Right*: Uncontrolled case with desired speed of 120km/h for each segment.

Table 6.5 for scenarios 1 through 4. Comparing to the case without control, our speed control strategy significantly reduces the fuel consumption amount on test highway. Meanwhile, the optimal solution can be obtained within average of 1.8 seconds using MATLAB installed in a standard desktop computer with a 3.50 GHz processor and a 16 GB RAM. The high computational performance indicates capability of real-time implementation. Therefore, the proposed method is verified to be applicable to a range of large scale real-world traffic control scenarios.

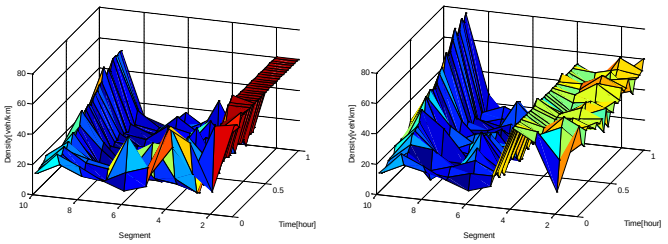

Figure 6.11 Density History of Scenario 4 for 6000 veh/h Demand at Origin. *Left*: Speed limit signs are controlled by the rounded optimal solution. *Right*: Uncontrolled case with desired speed of 120km/h for each segment.

Table 6.4 Comparison of TTT with and without control in high demanding profile ($\geq$ 5000 veh/h)

Scenarios	TTT w/o control [veh*h]	TTT with control [veh*h]	TTT Reduction Percentage
Demands:	217.08	196.84	9.32%
5000 veh/h	223.22	186.15	16.61%
	217.59	185.28	14.85%
	218.20	195.02	10.62%
	208.76	192.26	7.91%
			average: 11.86%
Demands:	228.65	186.65	18.37%
5500 veh/h	231.78	186.10	19.71%
	227.98	187.98	17.55%
	233.51	190.95	18.22%
	231.54	192.48	16.87%
			average: 18.14%
Demands:	232.48	189.45	18.51%
6000 veh/h	233.42	189.01	19.02%
	231.26	184.81	20.09%
	235.04	194.01	17.46%
	234.45	191.28	18.41%
			average: 18.70%

Table 6.5 Total fuel consumption in simulation scenarios with and without control

	with control	without control	reduction percentage	(with control - without control) t-test
Scenario 1:	$534.9kg$	$565.3kg$	5.37%	
4500	$563.5kg$	$583.3kg$	3.52%	
veh/h	$557.0kg$	$564.4kg$	1.32%	p-value = 0.0312
	$556.5kg$	$571.9kg$	2.69%	
	$564.9kg$	$594.1kg$	5.17%	95% CI: $(-38.50718, -2.37282)$
			average: 3.61%	
Scenario 2:	$579.2kg$	$622.6kg$	6.97%	
5000	$543.9kg$	$625.7kg$	13.06%	
veh/h	$541.7kg$	$618.8kg$	12.46%	p-value = 0.0002355
	$565.4kg$	$625.3kg$	9.57%	
	$561.6kg$	$605.9kg$	7.31%	95% CI: $(-80.60807, -41.99193)$
			average: 9.87%	
Scenario 3:	$542.9kg$	$648.0kg$	16.22%	
5500	$542.2kg$	$650.3kg$	16.62%	
veh/h	$552.8kg$	$640.3kg$	13.66%	p-value = 4.612e-08
	$557.1kg$	$649.3kg$	14.20%	
	$553.1kg$	$645.1kg$	14.26%	95% CI: $(-105.33765, -88.62235)$
			average: 14.99%	
Scenario 4:	$553.5kg$	$652.9kg$	15.22%	
6000	$547.1kg$	$648.0kg$	15.57%	
veh/h	$538.6kg$	$641.6kg$	16.05%	p-value = 1.89e-08
	$554.1kg$	$652.8kg$	15.12%	
	$554.5kg$	$645.2kg$	14.06%	95% CI: $(-107.36563, -89.71437)$
			average: 15.20%	

For each case, t-test is performed to statistically examine the performance of proposed optimal control strategy. The p-value and 95% confidence interval (CI) are shown in Table 6.5. The samples for taking t-test are fuel consumption resulting from five repeated simulations with different seed parameters. T-test for the difference of controlled and uncontrolled cases (5th column in Table 6.5) shows that p-value decreases when demanding volume increases. The decreasing trend demonstrates increasing reduction of fuel consumption compared with no controlled cases. Therefore, proposed control strategy is more effective when applied in sever congested scenarios. Furthermore, negative value of CI demonstrates the effectiveness of propose control strategy, i.e. fuel consumption is always reduced.

6.5 Distributed Traffic Speed Control for Travel Time Minimization

Using B-J/F solution to HJ Moskowitz function, we formulate a LP problem solved for travel time minimization. Unlike with previous investigation on optimal control of scalar conservation law, a distributed framework is constructed using a networked Road Infrastructures (RIs). Two distributed algorithms, projected subgradient method and ADMM, are incorporated in this distributed RIs network. Instead of relying on global information collection, distributed method only depends on local traffic information.

6.5.1 A General Formulation of Total Travel Time Minimization Problem

As illustrated in Fig. 6.1, a set of dynamic speed limits are controlled for travel time minimization on the highway mainstream. The travel time of all vehicles on all highway segments during time interval $[t_n, t_{n+1}]$ is expressed as

$$\begin{aligned} J &= \sum_{k=0}^{k_m-1} (Q_{up}^{(t_n,x_k)} - Q_{down}^{(t_n,x_k)})\Delta t \\ &= \sum_{k=0}^{k_m-1} J_k \end{aligned} \tag{5.40}$$

Constrained by ramp effect, travel time minimization can be formulated as a LP problem with respect to $Q_{up}^{(t_n,x_k)}$ and $Q_{down}^{(t_n,x_k)}$ as follows,

$$
\begin{aligned}
min.\ \ & J = (5.40) \\
s.t.\ \ & A_{model}^{(t_n,x_k)}\mathbf{y}^{(t_n,x_k)} \leq b_{model}^{(t_n,x_k)},\ k = 0, ..., k_m - 1 \\
& Q_{down}^{(t_n,x_k)} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if no ramp exists on } [x_k, x_{k+1}], \\
& (1 - R_{off}^{x_k})Q_{down}^{(t_n,x_k)} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if off-ramp exists on } [x_k, x_{k+1}], \\
& Q_{down}^{(t_n,x_k)} + C_{in}^{x_k} = Q_{up}^{(t_n,x_{k+1})},\ k = 0, ..., k_m - 1 \\
& \text{if on-ramp exists on } [x_k, x_{k+1}],
\end{aligned} \tag{5.41}
$$

$A_{model}^{(t_n,x_k)}\mathbf{y}^{(t_n,x_k)} \leq b_{model}^{(t_n,x_k)}$ are compact form of local model constraints for segment k at time t_n. In case of real-time traffic controller design, one-time-step optimization is carried out, which optimize the next speed limit based on current traffic flow and density information. This optimization process repeats for every time interval $[t_n, t_{n+1}]$ until $n = M - 1$. We omit the redefining of all parameters including A_{model}, b_{model}, $R_{off}^{x_k}$, $C_{in}^{x_k}$, as well as variable set $\mathbf{y}$ and keep them consistent with the definition in (4.32). In this case, $\mathbf{y}^{(t_n,x_k)}$ represent the local set of traffic inflow and outflow at t_n. Moreover, desired speed is calculated by following the same procedure in (4.33)-(4.35).

6.5.2 Applying Projected Subgradient Method in Distributed Traffic Control

In (5.41), we consider the ramp effect by adding affine equality constraints. It can also be incorporated in a compact form by using a fat and full rank matrix $A \in \mathbb{R}^{(M*k_m)\times(2*M*k_m)}$. It turns out the projection operator is also affine. Hence the update step simply follows the linear transformation as in (4.17) of section 3.4. We expand matrix A as $A =$

$\begin{bmatrix} A_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & A_{k_m} \end{bmatrix}$. which is a diagonal block matrix. Let $B = A^T(AA^T)^{-1}A$ and $B \in \mathbb{R}^{(2*M*k_m)\times(2*M*k_m)}$ is still a diagonal block matrix where only the entries associated with connected highway segments are non-zero. Therefore, updating step only relies on current local information $\mathbf{y}^{(t_n,x_{k'})}$, where k' is the index set of all segments connecting to the objective segment.

Another issue is that the local model constraints may not be satisfied if the primal set is projected to the affine set by (4.17) of section 3.4. In this case, we locally choose any violated model constraint, i.e. the local $A_{model}^{(t_n,x_k)}$. Then it is treated as the subgradient to update the state value. Therefore, state set recursively moves towards into the feasible region after each updating. If state set is feasible locally, we turn back to using projected updating step (4.17) until it violates the model constraints. We summarize modified projected subgradient method as follows.

Initialization: Initial Density at t_0, $n = 0$, $k = 0$;

while *less than the maximum iteration or not converged* **do**

  for *each highway segment at time* t_n *(in parallel)* **do**

    if $A_{model}^{(t_n,x_k)}\mathbf{y}^{(t_n,x_k)} \leq b_{model}^{(t_n,x_k)}$ **then**

      $\mathbf{y}^{(t_n,x_k)} = \mathbf{y}^{(t_n,x_k)} - \alpha(I - A_k^T(A_kA_k^T)^{-1}A_k)\frac{\partial J_k}{\partial \mathbf{y}^{(t_n,x_k)}}$

    end

    else

      $g(\mathbf{y}^{(t_n,x_k)}) = A_{model}^{(t_n,x_k)}$

      $\mathbf{y}^{(t_n,x_k)} = \mathbf{y}^{(t_n,x_k)} - \alpha(I - A_k^T(A_kA_k^T)^{-1}A_k)g(\mathbf{y}^{(t_n,x_k)})^T$

    end

  end

end

Protocol 7 Solving for Travel Time Minimization Problem with Projected Subgradient Method

6.5.3 Applying ADMM in Distributed Traffic Control

In the application of ADMM for travel time minimization, we sequentially find the optimal solution during each time interval. A general form of augmented Lagragian at any time t_n is built as follows. We ignore the time notation t_n for simplicity in the following equations.

$$L(\mathbf{y}, \boldsymbol{\mu}) = \sum_{k=0}^{K} f^k(\mathbf{y}^{x_k}) + \boldsymbol{\mu}^T(A\mathbf{y}^{x_k} - b) + \frac{\rho}{2}||A\mathbf{y}^{x_k} - b||_2^2 \tag{5.42}$$

where A and b are parameter matrix and vector to represent ramp effect. By dual decomposition, each subproblem is solved by minimizing the following Lagrangian

$$\begin{aligned} min. \quad & L_k(\mathbf{y}^{x_k}, \mathbf{y}^{x_S}, \boldsymbol{\mu}_{local}) = f^k(\mathbf{y}^{x_k}) + \boldsymbol{\mu}_{local}^T\mathbf{y}^{x_k} + (\mathbf{y}^{x_k})^T H^k \mathbf{y}^{x_k} \\ s.t. \quad & A_{model}^{x_k}\mathbf{y}^{x_k} \leq b_{model}^{x_k} \end{aligned} \tag{5.43}$$

where in case of $k = 2, ..., k_m - 1$

$$\boldsymbol{\mu}_{local} = \begin{cases} [-\mu_{k-1} - \rho(1 - R_{off}^{x_k})Q_{down}^{x_{k-1}},\ \mu_k]^T, \\ \text{if off-ramp off-ramp exits on} [x_{k-1}, x_k] \\ [-\mu_{k-1} - \rho(Q_{down}^{x_{k-1}} + C_{in}^{x_k}),\ \mu_k]^T, \\ \text{if on-ramp off-ramp exits on} [x_{k-1}, x_k] \\ [-\mu_{k-1},\ \mu_k]^T, \\ \text{if no ramp exits on} [x_{k-1}, x_k] \end{cases} \tag{5.44}$$

For $k = 0$ one has

$$\boldsymbol{\mu}_{local} = \begin{cases} [0,\ \mu_0(1 - R_{off}^{x_k})]^T, \\ \text{if off-ramp exits on} [x_0, x_1] \\ [0,\ \mu_0]^T, \\ \text{if on-rampor no ramp exits on} [x_0, x_1] \end{cases} \tag{5.45}$$

Finally for $k = k_m$, one has

$$\boldsymbol{\mu}_{local} = \begin{cases} [-\mu_{k_m-1} - \rho(1 - R_{off}^{x_k})Q_{down}^{x_{k_m-1}},\ 0]^T, \\ \text{if off-ramp exits on} [x_{k_m-1}, x_{k_m}] \\ [-\mu_{k_m-1} - \rho(Q_{down}^{x_{k_m-1}} + C_{in}^{x_{k_m}}),\ 0]^T, \\ \text{if on-ramp exits on} [x_{k_m-1}, x_{k_m}] \\ [-\mu_{k_m-1},\ 0]^T, \\ \text{if no ramp exits on} [x_{k_m-1}, x_{k_m}] \end{cases} \tag{5.46}$$

Hessian matrix H^k is diagonal which is defined as

$$H^k = \begin{bmatrix} 1 & 0 \\ 0 & 1 - R_{off}^{x_k} \end{bmatrix}$$

We first saparetly solve (5.43) and apply optimal solutions from each optimizer as the initial value $\mathbf{y}_0^{x_k}$. Then follow ADMM iterations in the Protocol shown in section 3.5 to

sequentially obtain $\mathbf{y}_q^{x_k}$ for each segment. Solving for each subproblem relies on the newly updated solution on segment $k-1$ at iteration q, i.e. $\mathbf{y}_q^{x_{k-1}}$, as well as the last updated solution on segment $k+1$, i.e. $\mathbf{y}_{q-1}^{x_{k+1}}$. By completing state iteration in a sequential fashion, we update Lagrangian multiplier μ using local state information. This procedure repeats at each time instant t_n until t_{M-1}.

Comparing with dual subgradient method, ADMM updates state variables in sequential fashion, which means $\mathbf{y}_p^{x_{k+1}}$ depends on current optimal value of $\mathbf{y}_q^{x_k}$. While dual subgradient method can be implemented in parallel sense. However, ADMM is still effective and efficient in solving this problem due to the linearity of the objective. In ADMM, augmented Lagrangian becomes strict convex by adding quadratic term in (5.42). It guarantees the convergence of primal solution while it does not always hold using dual subgradient method. In practice, we always find the better performance of ADMM on convergence and non-sensitivity of stepsize parameter ρ. In numerical example section, we adopt ADMM to achieve a modest accuracy within tens of iterations.

6.5.4 Simulation and Discussion

In simulation, we adopt the same test highway scenario and the layout of speed limit signs as in section 6.4.4. To compare ADMM based distributed algorithm in solving traffic travel time minimization problem, we conduct a set of simulations using different optimization approach. First, we directly solve original problem without dual decomposition in a centralized manner. Then, subproblems are solved by projected subgradient method and ADMM. The last, no control case is taken into account by setting an identical constant speed limit for each segment. Demand traffic flow is $4500veh/h$ and remain constant during $2h$ simulation time. We record inflow and outflow vehicle count for every $2min$ and calculate the total number of vehicles on test highways section. Time history of total vehicles number (vehicle conservations) is plotted in Fig.6.12.

Speed limit control is carried out after the the traffic get to the stable state. Hence

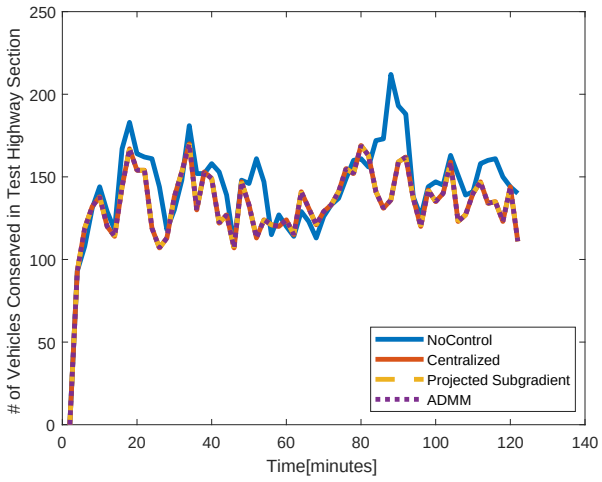

Figure 6.12 Time History of Vehicles Conserved in test Highway Section

there is no difference prior to $2min$. We notice that consistent vehicles counts are recorded using centralized and distributed optimization algorithm, which numerically verify the effectiveness of distributed method. To save space, we extract the history of iterations at the first time instance in Fig. 6.13 and 6.14. Both of them demonstrate the convergence to global optimal solution using projected subgradient method and ADMM. f^* and y^* are optimal objective value and associated optimal solution obtained by solving 5.41 using LP solver in centralized fashion. Although both two types of distributed algorithm will finally converge to the optimal point, ADMM provides a better performance than projected gradient method. First ADMM takes tens of iterations for the convergence. Moreover, there is one non-sensitive stepsize parameter in ADMM. However, two sensitive stepsize parameters are required to determined in projected gradient method. However, we notice that they result in a overlapping results of reduced vehicle counts shown in Fig.6.13. We make further comparison regarding total travel time in Table.6.6.

Comparing with no controlled scenario with different fixed speed limit settings, both centralized and distributed method are numerically verified effective in reducing total travel time. Since either projected gradient method or ADMM equivalently affects the travel

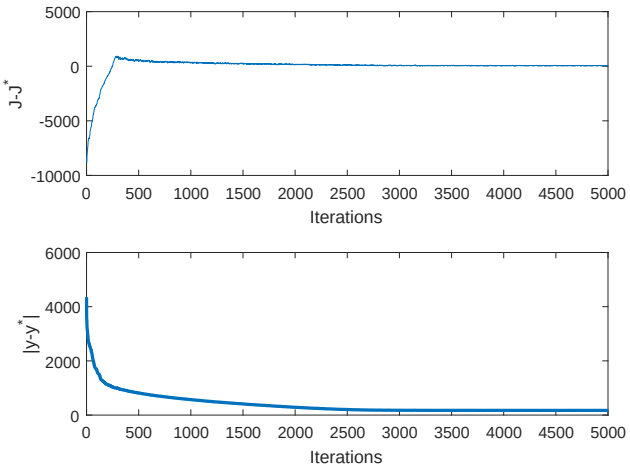

Figure 6.13 Iteration history by projected subgradient method. *Upper*: The difference of objective value at each iteration and at $\mathbf{y}^*$. *Lower*: l^2 norm of $\mathbf{y} - \mathbf{y}^*$ at each iteration.

Table 6.6 Comparison of TTT

TTT without control	TTT with centralized optimizer	TTT with distributed optimizer	reduction percentage
291.80 $veh * h$	270.03 $veh * h$	270.03 $veh * h$	7.46%

time, we do not distinguish them in Table.6.6. We notice that increasing speed limit will not reduce the travel time in case of high traffic demand. Reversely, more vehicles are conserved in test highway section by simply setting a higher speed limit.

6.6 A MIQQ Based Real-Time Control Strategy for Highway Travel Time Minimization

The traffic management and control design for travel time minimization is extended to a highway network scenario in this application. Different from the controller designed in previous examples where only dynamic speed limits are implemented, we develop an efficient strategy for controlling the hybrid highway infrastructures including dynamic speed limits,

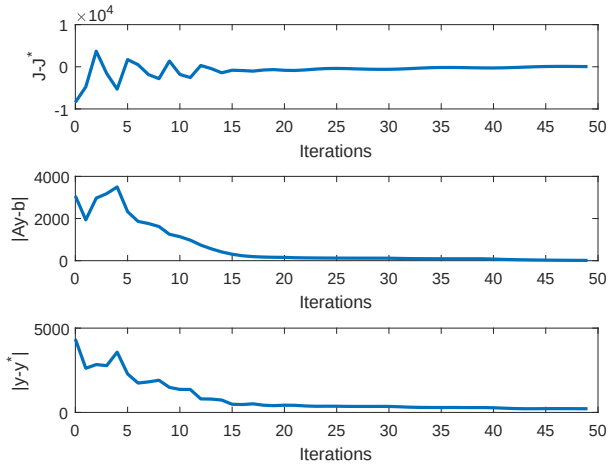

Figure 6.14 Iteration history by ADMM. *Upper*: The difference of objective value at each iteration and at $\mathbf{y}^*$. *Middle*: l^2 norm of $A\mathbf{y} - b$ at each iteration. *Lower*: l^2 norm of $\mathbf{y} - \mathbf{y}^*$ at each iteration.

ramp metering and information board. The traffic density is predicted based on the flow dynamic model and corrected periodically by measured traffic flow data. The minimum travel time traffic control problem is then formulated as MIQQ. Numerical simulation of a real world highway network is provided to demonstrate significant reduction of TTT and alleviation of traffic congestion compared to results obtained from ALINEA and PI-ALINEA methods.

6.6.1 Highway Network

6.6.1.1 Components of a Highway Network

A highway network is composed of nodes and edges that connect two distinct nodes, as illustrated in Fig. 6.15. Each node $n \in \mathcal{N}$ represents one of the three types of location:

1. The conjunction of different highway mainstreams, specifically, the conjunction node $n_c^i \in \mathcal{N}_{c_{out}}$, e.g., n_c^2 and n_c^4 in Fig. 6.15, that allows for outgoing traffics, or $n_c^j \in \mathcal{N}_{c_{in}}$, e.g., n_c^1 and n_c^5, that have incoming traffic flow from other highway sections, where $\mathcal{N}_{c_{out}}$ and $\mathcal{N}_{c_{in}}$ are two subsets of $\mathcal{N}$.

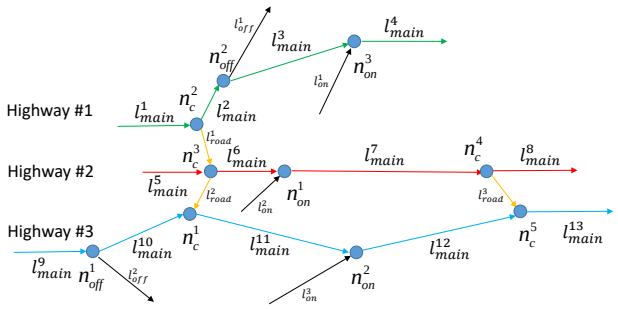

Figure 6.15 Example of a highway network. Mainstream links are marked as green for Highway #1, red for Highway #2, and blue for Highway #3. *Black arrows*: on-ramps or off-ramps connecting urban/rural roadway to highway. *Yellow arrows*: roadway links connecting different highways links.

2. The joint where incoming traffic contributes to the highway mainstream from on-ramp, denoted as $n_{on}^{i} \in \mathcal{N}_{on}$.

3. The joint where traffic exits the mainstream via off-ramp, denoted as $n_{off}^{i} \in \mathcal{N}_{off}$.

Based on the above assumptions, the set $\mathcal{N}$ contains four subsets, $\mathcal{N} = \{\mathcal{N}_{c_{in}}, \mathcal{N}_{c_{out}}, \mathcal{N}_{on}, \mathcal{N}_{off}\}$. Similarly, three types of edges are defined below:

1. Mainstream link (highway link), denoted by $l_{main}^{m} \in \mathcal{L}_{main}$, $m = 1, ..., L_{main}$, that connects two adjacent nodes to form the mainstream of traffic on highway section, e.g. l_{main}^{7} represent highway link (n_{on}^{1}, n_{c}^{4}) in Fig. 6.15.

2. On-ramp or off-ramp, denoted as $l_{on}^{h} \in \mathcal{L}_{on}$, $h = 1, ..., L_{on}$ and $l_{off}^{g} \in \mathcal{L}_{off}$, $g = 1, ..., L_{off}$, respectively, that allows the traffic entering or exiting the highway.

3. Roadway, denoted by $l_{road}^{r} \in \mathcal{L}_{road}$, $r = 1, ..., L_{road}$, represents the link for which traffic flow changes the route from one highway section to another, e.g. l_{road}^{3} represents roadway link (n_{c}^{4}, n_{c}^{5}) in Fig. 6.15.

The total number of links is determined by $L = L_{main} + L_{road} + L_{on} + L_{off}$. Furthermore, we make the following assumptions to simplify the problem.

Assumption 6.6.1. *The on-ramp traffic volume $Q_{on}^{(t_p, l_{on}^{h})}$, is constant and controlled by ramp metering.*

Assumption 6.6.2. *The off-ramp vehicle volume is proportional to the corresponding downstream volume at mainstream link* l^m_{main} *with the constant ratio* $R^{(t_p,l^g_{off})}_{off}$ *during* $[t_p, t_{p+1}]$.

Assumption 6.6.3. *The traffic flow exiting a highway section via roadway link* l^r_{road} *remains a constant ratio* $R^{(t_p,l^r_{road})}_{off}$ *with respect to the corresponding downstream volume at mainstream link* l^m_{main} *during* $[t_p, t_{p+1}]$. *The downstream flow of link* l^r_{road}, *denoted by* $Q^{(t_p,l^r_{road})}_{on}$, *is controlled by a ramp meter.*

6.6.1.2 A Hybrid Traffic Control Infrastructure

A hybrid traffic control infrastructure, consisting dynamic speed limit sign, ramp metering, and highway information board, is expected to improve efficiency of the traffic management than a single control method. As illustrated in Fig.6.16, the dynamic speed

Figure 6.16 Example of Hybrid Control Infrastructures.

limit sign is employed as one of the traffic management infrastructures to control flow on each highway link. The desired volume can be obtained by displaying an appropriate speed $v(t_p, l^m_{main})$ on the speed limit sign. According to the time-varying traffic states, the speed limit on each mainstream link l^m_{main} is adjusted periodically via the control variable set $\mathbf{v}$=$[v(t_0, l^1_{main}), ..., v(t_0, l^{L_{main}}_{main}), \ldots, v(t_{P-1}, l^1_{main}), ..., v(t_{P-1}, l^{L_{main}}_{main})]$.

The ramp metering controls the outflow of on-ramp traffic, $Q^{(t_p,l)}_{on}$ for all $l \in \{\mathcal{L}_{road}, \mathcal{L}_{on}\}$. The one vehicle per green principle is adopted in meter control, where one vehicle is allowed to pass the meter during a short green light cycle. To obtain the desired on-ramp volume, the meter cycle length $\mathbf{T}$ is designed for each time interval at downstream of l^r_{road} and l^h_{on}.

The control variable set for ramp metering is $\mathbf{T} = [T(t_0, l_{on}^1), ..., T(t_0, l_{on}^{L_{on}}), T(t_{P-1}, l_{on}^1), ..., T(t_{P-1}, l_{on}^{L_{on}}), T(t_0, l_{road}^1), ..., T(t_0, l_{road}^{L_{road}}), T(t_{P-1}, l_{road}^1), ..., T(t_{P-1}, l_{road}^{L_{road}})]$.

The highway information board guides the traffic to their destination by selecting the optimal routes. For example, as shown in Fig. 6.15, traffic from Highway #2 with destination n_c^5 is guided to travel via l_{road}^2 or l_{road}^3. The highway information board is located at link l_{main}^5 in this case. The control variable set determining the route selection is set as $\mathbf{b} = [b^{(t_0, l_{road}^1)}, ..., b^{(t_0, l_{road}^{L_{road}})}, ..., b^{(t_{P-1}, l_{road}^1)}, ..., b^{(t_{P-1}, l_{road}^{L_{road}})}]$. The element in $\mathbf{b}$ is defined as a binary variable according to

$$b^{(t_p, l_{road}^r)} = \begin{cases} 1, \text{ if } l_{road}^r \text{ is allowed} \\ 0, \text{ if } l_{road}^r \text{ is not allowed.} \end{cases} \tag{6.47}$$

In practice, the highway information board closes or activates the links between highway sections. If a link is closed, alternative route information will be displayed on the information board. For example, if l_{road}^2 is closed, then traffic with destination n_c^5 will be guided to travel trough l_{road}^3.

6.6.2 Problem Formulation

6.6.2.1 Intermediate Control Variables and The Objective Function

To minimize the TTT of the highway network, the minimum time traffic management problem is formulated as a MIQQ problem. The intermediate control variables include upstream, $Q_{up}^{(t_p, l_{main}^m)}$, and downstream traffic flow, $Q_{down}^{(t_p, l_{main}^m)}$, of all highway links, the outflow at the end of all on-ramps, $Q_{on}^{(t_p, l_{road}^r)}$ and $Q_{on}^{(t_p, l_{on}^h)}$, as well as the binary variables for route selection, $b^{(t_p, l_{road}^r)}$, during each time interval $[t_p, t_{p+1}]$ for all $p = 0, \ldots, P-1$. Therefore, the intermediate control variables include both continuous and binary variable sets.

The TTT for all vehicles in the highway network over the duration $[t_1, t_P]$ consists of traversing time J_m along all highway link l^m_{main}, waiting time J_c in the queue on all roadway link l^r_{road}, as well as the waiting time J_{on} on all on-ramp l^h_{on}. Accordingly, the objective is expressed as

$$\begin{aligned} J=&J_m + J_c + J_{on} \\ =&\sum_{p=0}^{P-1} \Delta t\{ \sum_{l^m_{main}\in\mathcal{L}_{main}} [(Q^{(t_p,l^m_{main})}_{up} - Q^{(t_p,l^m_{main})}_{down})\Delta t \\ &+\rho^{(t_p,l^m_{main})}_{ini} X_{l^m_{main}}] + \sum_{l^r_{road}\in\mathcal{L}_{road}} [(Q^{(t_p,l^r_{road})}_{up} \\ &-Q^{(t_p,l^r_{road})}_{on})\Delta t + \rho^{(t_p,l^r_{road})}_{ini} X_{l^r_{road}}] \\ &+ \sum_{l^h_{on}\in\mathcal{L}_{on}} [(Q^{(t_p,l^h_{on})}_{up} - Q^{(t_p,l^h_{on})}_{on})\Delta t + \rho^{(t_p,l^h_{on})}_{ini} X_{l^h_{on}}]\}, \end{aligned} \tag{6.48}$$

where $X_{l^m_{main}}$, $X_{l^r_{road}}$, and $X_{l^h_{on}}$ are the segment length of link l^m_{main}, l^r_{road}, and l^h_{on} respectively, ρ_{ini} is the initial density and periodically updated by the new measurements from the volume sensors, and $Q^{(t_p,l^h_{on})}_{up}$ is obtained from the volume sensors. $Q^{(t_p,l^r_{road})}_{up}$ is a quadratic function of the intermediate variables, expressed as

$$Q^{(t_p,l^r_{road})}_{up} = Q^{(t_p,l^m_{main})}_{down} b^{(t_p,l^r_{road})} R^{(t_p,l^r_{road})}_{off} + Q_{other}$$

where traffic on link l^m_{main} flows into other highway links via l^r_{road} and Q_{other}, measured by volume sensors, is the volume contributed from other resources excluding l^m_{main}.

6.6.2.2 MIQQ Problem Formulation

By incorporating the linear model constraints describing the traffic dynamics of each highway into the objective function, the hybrid traffic control problem to minimize the TTT of the highway network is formulated as

$$
\begin{aligned}
&\min_{\mathbf{y}} J = (6.48)\\
&s.t.(a)\ A_{model}\mathbf{y} \le b_{model}\\
&\quad (b)\ 0 \le (Q_{up}^{(t_p,l^r_{road})} - Q_{on}^{(t_p,l^r_{road})})\Delta t + \rho_{ini}^{(t_p,l^r_{road})} X_{l^r_{road}}\\
&\qquad < \mathcal{MAX}_{l^r_{road}}, \forall\ l^r_{road} \in \mathcal{L}_{road}, p = 0, ..., P-1,\\
&\quad (c)\ 0 \le (Q_{up}^{(t_p,l^h_{on})} - Q_{on}^{(t_p,l^h_{on})})\Delta t + \rho_{ini}^{(t_p,l^h_{on})} X_{l^h_{on}}\\
&\qquad < \mathcal{MAX}_{l^h_{on}}, \forall\ l^h_{on} \in \mathcal{L}_{on}, p = 0, ..., P-1,\\
&\quad (d)\ (1 - R_{off}^{(t_p,l)}) Q_{down}^{(t_p,l^j_{main})} = Q_{up}^{(t_p,l^k_{main})},\ p = 0, ..., P-1\\
&\qquad \text{if } l^j_{main}\ l^k_{main} \text{and } l \text{ are separated at } n^i_{off} \in \mathcal{N}_{off}\\
&\qquad \text{or } n^q_c \in \mathcal{N}_{c_{out}},\\
&\qquad l^j_{main}, l^k_{main} \in \mathcal{L}_{main},\ l \in \{\mathcal{L}_{off}, \mathcal{L}_{road}\},\\
&\quad (e)\ Q_{down}^{(t_p,l^j_{main})} + Q_{on}^{(t_p,l)} = Q_{up}^{(t_p,l^k_{main})},\ p = 0, ..., P-1\\
&\qquad \text{if } l^j_{main}\ l^k_{main} \text{and } l \text{ are separated at } n^i_{on} \in \mathcal{N}_{on}\\
&\qquad \text{or } n^q_c \in \mathcal{N}_{c_{in}},\\
&\qquad l^j_{main}, l^k_{main} \in \mathcal{L}_{main},\ l \in \{\mathcal{L}_{on}, \mathcal{L}_{road}\},
\end{aligned}
\tag{6.49}
$$

where $\mathbf{y}$ represent the intermediate control variable set. The linear model constraints describing the traffic dynamics in (3.22) are represented in a compact form in (a) of (6.49). To prevent traffic congestion on ramps due to ramp metering, additional quadratic and linear inequality constraints are introduced in (b) and (c). The left hand side of (b) and (c) compute the number of vehicles on links l^r_{road} and l^h_{on} for every time interval and are assumed to be less than the pre-determined maximum allowed vehicle number $\mathcal{MAX}_{l^r_{road}}$

and $\mathcal{MAX}_{l^h_{on}}$, respectively. According to assumptions 6.6.1-6.6.2, equalities (d) and (e) hold in (6.49). Since the above formulation includes mixed continuous and binary variables and the objective is a quadratic function subject to quadratic and linear constraints, problem (6.49) is classified as a MIQQ problem.

6.6.2.3 Conversion from Intermediate Control Variable Set to Final Set

Solution from problem (6.49) generates intermediate control variables. To convert the intermediate control variables to the final ones, including $\mathbf{v}$, $\mathbf{T}$, and $\mathbf{b}$, to display on the traffic infrastructures, the following transformations are required.

First, based on the triangular fundamental diagram defined in (1.1), elements in $\mathbf{v}$ are determined by

$$v(t_p, l^m_{main}) = \begin{cases} v_f^{l^m_{main}}, \text{ if } 0 \leq \rho \leq \rho_c^{l^m_{main}} \\ w^{l^m_{main}}(1 - \dfrac{\rho_j^{l^m_{main}}}{\rho}), \text{ if } \rho_c^{l^m_{main}} < \rho \leq \rho_j^{l^m_{main}} \end{cases} \tag{6.50}$$

where

$$\rho = \frac{(Q_{up}^{(t_{p-1}, l^m_{main})} - Q_{down}^{(t_{p-1}, l^m_{main})})\Delta t}{X_{l^m_{main}}} + \rho_{ini}^{(t_{p-1}, l^m_{main})}. \tag{6.51}$$

Hence, the parameters in the fundamental diagram of highway link l^m_{main} are represented by $v_f^{l^m_{main}}$, $w^{l^m_{main}}$, $\rho_c^{l^m_{main}}$ and $\rho_j^{l^m_{main}}$ in (6.50)-(6.51).

Second, the meter cycle length $\mathbf{T}$ is set in unit of second and calculated based on the outflow of on-ramp in form of. Given a constant green phase T_g that allows one vehicle passing the meter during that cycle, the controlled meter cycle length is determined by

$$T(t_p, l) = \frac{T_g * 3600}{Q_{on}^{(t_p, l)}}, \ l \in \{\mathcal{L}_{road}, \mathcal{L}_{on}\}. \tag{6.52}$$

As the final control variable set $\mathbf{b}$ is consistent with the corresponding ones in the intermediate control variables, no conversion is required for this set.

6.6.2.4 Implementation of the Hybrid Traffic Control Strategy

When determining control variables, the traffic flow and density are predicted based on current density and the traffic flow dynamics model. However, the traffic dynamic model is not an ideal one due to uncertainties of the fundamental diagram. In order to improve prediction accuracy, the real world traffic flow data is measured to update real-time density periodically. During each updating period, the control system integrates the sensing-optimizing-displaying (SOD) procedure which is illustrated in Fig.6.17.

The volume sensors record the amount of vehicles entering and fluxing out of all highway links l^m_{main}, roadway links l^r_{road} and on-ramp links l^h_{on} during the updating period $[t_{p-p'}, t_p]$. p' is defined as time steps in between two consecutive updating times. After receiving the measured volume data, the traffic density is updated via

$$\rho_{ini}^{(t_p,l)} = \frac{N_{up}^{(t_{p-p'},l)} - N_{down}^{(t_{p-p'},l)}}{X_l} + \rho_{ini}^{(t_{p-p'},l)}, \tag{6.53}$$

where $l \in \{\mathcal{L}_{main}, \mathcal{L}_{road}, \mathcal{L}_{on}\}$, $N_{up}^{(t_{p-p'},l)}$ and $N_{down}^{(t_{p-p'},l)}$ are measured number of vehicles during $[t_{p-p'}, t_p]$ at upstream and downstream of link l. Before new measurements from the volume sensors become available at the next updating time $t_{p+p'}$, problem (6.49) will be solved at each time interval t_{p+i}, t_{p+i+1} for index i and $0 \leq i < p'$ to determine the new intermediate control variables which are converted into the final control variables based on (6.50)-(6.52). Values of the final control variables at each time interval are then sent to corresponding dynamic speed limit signs, ramp metering and highway information boards. The density will be updated by new measured data at $t_{p+p'}$, which initiates the next SOD procedure.

6.6.3 Simulation and Discussion

6.6.3.1 A Highway Network Example and VISSIM Settings

In this section, a real world scenario is considered as a test highway network. We extract two sections with 6.08 km length of each one from two major highways, I-35 and US-69,

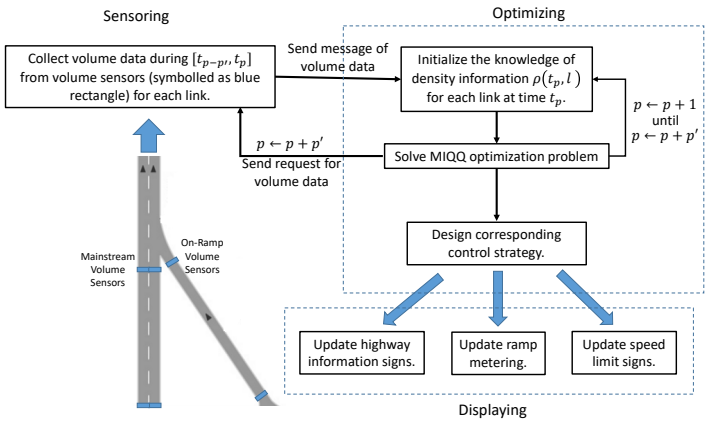

Figure 6.17 A Sensoring-Optimizing-Displaying (SOD) Procedure for Real-Time Highway Traffic Control

which are located between the cities Ames and Des Moines in Iowa, as illustrated in Fig. 6.18. Two on-ramps divide the test highway sections into four segments. Two roadways, with one located in north and another one in south, allows traffic traveling between US-69 and I-35.

Volume sensors are installed at the starting and ending point of each highway segment, roadway link and on-ramp. Each of them records the number of vehicle entering and fluxing out of the corresponding link. For every 8 mins, the volume sensors send the sensor data to the computation center update real time density on highway segments and on-ramps.

VISSIM is connected to the MIQQ solver [K. Holmstrom and Edvall (2005)] through COM interface in MATLAB. To control speed limit in this study, the desired speed limits are dynamically adjusted at upstream point of each highway segment for every 2 mins. Moreover, ramp metering cycle lengths are updated per 2 mins based on the optimal solution from MIQQ. For ramps that are supposed to be closed from the control results, traffic volume of the corresponding ramp is assumed to be zero in the simulation.

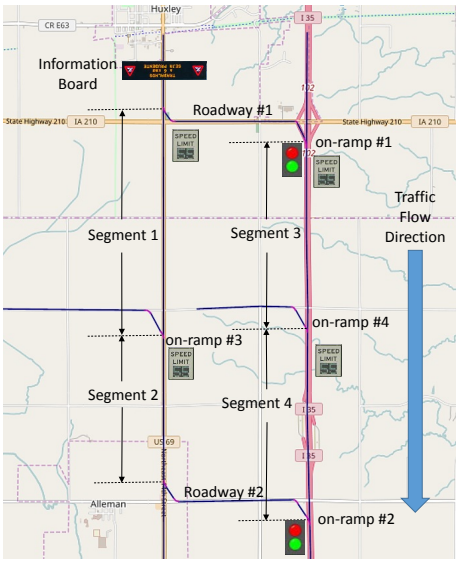

Figure 6.18 A Test Highway Network

6.6.3.2 Simulation Results

In order to compare the performance of MIQQ with other control strategies, four typles of simulation results are provided, including cases without control, ALINEA strategy, PI-ALINEA ramp metering method, and the proposed MIQQ method. Relative setting from the four methods are shown in Table 6.7. For each case, the simulation lasts for 3 hours, where simulation from the 1st 20 mins is ignored due to the unstable traffic status at the beginning time.

ALINEA is a popular local responsive feedback ramp metering strategy, and has been verified to be an effective strategy in both field tests and simulation [Hadj-Salem et al. (1990); Papageorgiou et al. (1997)]. ALINEA determines the metering rates based on the downstream mainline occupancy from the meter. Its objective is to maximize the mainline throughout by maintaining occupancy values below the preset threshold. Since it focuses on preventing merging congestion, ALINEA requires the real-time occupancy measurements around the merging areas to achieve efficiency. However, bottlenecks may be further away from the merging areas in real world scenarios, where ALINEA cannot lead to high efficiency.

Thus, PI-ALINEA, a Proportional-Integral (PI) extension of ALINEA has been proposed and proved to be an efficient ramp-metering algorithm in the presence of far-downstream bottlenecks [Kan et al. (2016)].

Table 6.7 Comparison of traffic control strategies

	No Control	ALINEA	PI-ALINEA	MIQQ
Speed Limit Signs	90km/h	90km/h	90km/h	Dynamic
Ramp Metering	Unavailable	Dynamic	Dynamic	Dynamic
Information Board	Unavailable	Unavailable	Unavailable	Available
TTT $[veh * h]$	633.26	634.53	624.53	542.63

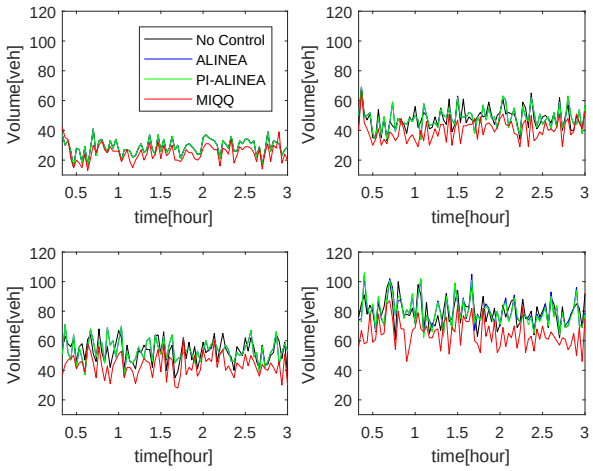

Figure 6.19 Time history of vehicle conservation at highway segment 1 (*upper left*), highway segment 2 (*upper right*), highway segment 3 (*lower left*) and highway segment 4 (*lower right*).

To verify the feasibility of the proposed MIQQ strategy in high traffic demands, north-to-south traffic flow are set to 1600 veh per hour (vph) and 1200 vph at source location of highway I-35 and US-69, respectively. Traffic volumes is 600 vph for on-ramp #3 and #4.

It is assumed that 20% of traffic flow on US-69 coming from north will transfer to I-35. During each time interval, they are guided to travel through roadway #1 or #2 by the highway information board. For every 8 mins, densities are updated by the measured data on each highway link and on-ramp. The control variables are regenerated and displayed through the hybrid infrastructures every 2 mins.

Simulation results are shown in Table 6.7 and Fig. 6.19. The proposed MIQQ leads to further reduced TTT compared to the other three methods. The TTT reduction percentages are 14.31%, 14.48% and 13.11% compared to cases with no control, ALINEA, and PI-ALINEA, respectively. Furthermore, less vehicles are observed in each test highway link for every time interval. The comparative results verify that the proposed MIQQ strategy has improved efficiency in congestion alleviation during rush hours.

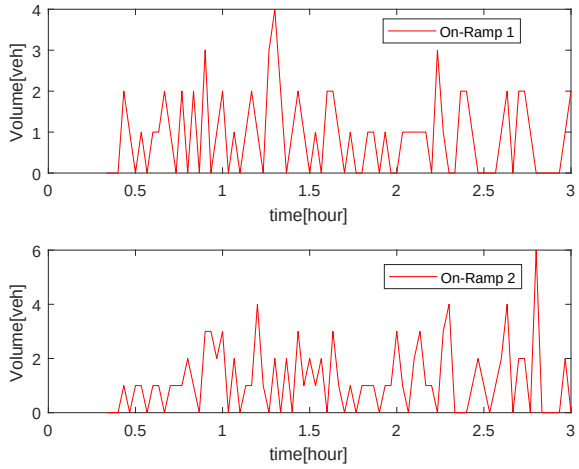

Figure 6.20 Time history of vehicle conservation at on-ramp 1 (*upper*) and 2 (*lower*).

Since not all on ramps have ramp metering in a real world highway network, we assume only a subset of on-ramps is controlled by ramp metering. In this case, on-ramp 1 and 2 are controlled while on-ramp 3 are 4 have no ramp metering. Time history of vehicle count of on-ramps 1 and 2 are illustrated in Fig. 6.20. A threshold is considered to restrict queue length at on-ramps, i.e. maximum 6 vehicles in the waiting queue. Figure 6.20 demonstrates the queue length restriction is satisfied on each of the two controlled on-ramps.

6.7 Conclusion

To mathematically model the traffic dynamics, we design a new explicit affine expressions for describing traffic flow in any future time on highway section. Based on Lighthill-Whitham-Richard partial differential equation, Cauchy problem is modeled as an optimization problem. The explicit solution to Cauchy problem is derived based on the Lax-Hopf formula and Greenshields Fundamental Diagram. Affine model constraints are considered in the Barron-Jensen/Frankowska solution. Traffic management problem can be formulated as convex/non-convex optimization problem by incorporating affine model constraints.

In the first application, we propose an efficient convex optimization problem formulation for fuel consumption minimization. After modeling the performance index as a quadratic function, the real-time fuel-efficient traffic control problem is formulated as a convex optimization problem. Simulation results demonstrate the reduced fuel consumption and alleviated traffic congestion. The feasibility of proposed optimization method is verified through VISSIM simulation in which different traffic volume and random seed parameters are considered.

In the second application, travel time is minimized by LP in distributed manner. We design two efficient local optimizers based on projected subgradient method and ADMM, respectively. Numerical simulation shows a consistent result obtained by using centralized method, which shows reduced TTT on test highway section. Comparing with projected subgradient method, it takes much less iterations of convergence for ADMM. Besides, ADMM requires only one step size design while two of them are necessary in projected subgradient method. Hence, we conclude ADMM is better than projected subgradient method in terms of implementation simplicity and iterative efficiency.

In the last application, we extend to solving a non-convex problem for traffic management in a highway network. A time efficient traffic control strategy is presented by using hybrid highway infrastructures, including dynamic limit signs, ramp metering, and highway information boards. The minimum time transportation problem for the entire highway net-

work is formulated as MIQQ. Performance of the proposed MIQQ method is verified in a real world simulation example using VISSIM. Compared to existing methods, ALINEA and PI-ALINEA, MIQQ lead to more reduced travel time and alleviation of congestion during busy hours.

BIBLIOGRAPHY

Acikmese, B., Scharf, D., Carson, J. M., and Hadaegh, F. Y. (2008). Distributed estimation for spacecraft formations over time-varying sensor topologies. In *IFAC World Conference*.

Adeli, H. and Cheng, N.-T. (1994). Augmented lagrangian genetic algorithm for structural optimization. *Journal of Aerospace Engineering*, 7(1):104–118.

Aspragathos, N. A. and Dimitros, J. K. (1998). A comparative study of three methods for robot kinematics. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 28(2):135–145.

Aubin, J.-P., Bayen, A. M., and Saint-Pierre, P. (2008). Dirichlet problems for some hamilton–jacobi equations with inequality constraints. *SIAM Journal on Control and Optimization*, 47(5):2348–2380.

Awad, Z. K., Aravinthan, T., Zhuge, Y., and Gonzalez, F. (2012). A review of optimization techniques used in the design of fibre composite structures for civil engineering applications. *Materials & Design*, 33:534–544.

Barkenbus, J. N. (2010). Eco-driving: An overlooked climate change initiative. *Energy Policy*, 38(2):762–769.

Barnes, M., Leather, H., and Arvind, D. (2007). Emergency evacuation using wireless sensor networks. In *Local Computer Networks, 32nd IEEE Conference on*, pages 851–857.

Barron, E. and Jensen, R. (1990). Semicontinuous viscosity solutions for hamilton–jacobi

equations with convex hamiltonians. *Communications in Partial Differential Equations*, 15(12):293–309.

Bayen, A. M., CLAUDEL, C., and SAINT-PIERRE, P. (2007). Computation of solutions to the moskowitz hamilton-jacobi-bellman equation under viability constraints. In *Decision and Control, 46th IEEE Conference on*, pages 4737–4742.

Bejan, A. and Lorente, S. (2001). Thermodynamic optimization of flow geometry in mechanical and civil engineering. *Journal of Non-Equilibrium Thermodynamics*, 26(4):305–354.

Bellman, R. (1958). On a routing problem. *Quarterly of applied mathematics*, pages 87–90.

Berkelaar, A., Dert, C., Oldenkamp, B., and Zhang, S. (2002). A primal-dual decomposition-based interior point approach to two-stage stochastic linear programming. *Operations Research*, 50(5):904–915.

Bertsekas, D. P. and Scientific, A. (2015). *Convex optimization algorithms*. Athena Scientific Belmont.

Bertsekas, D. P. and Tsitsiklis, J. N. (1989). *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ.

Bhaskar, V., Gupta, S. K., and Ray, A. K. (2000). Applications of multiobjective optimization in chemical engineering. *Reviews in Chemical Engineering*, 16(1):1–54.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. (2005). Gossip algorithms: Design, analysis and applications. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1653–1664. IEEE.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Boyd, S. and Vandenberghe, L. (2004, pp. 488). *Convex optimization*. Cambridge University Press, Cambridge.

Boyd, S., Xiao, L., and Mutapcic, A. (2003). Subgradient methods. *lecture notes of EE392o, Stanford University, Autumn Quarter*.

Bryson, A.E.and Ho, Y. (1975). *Applied Optimal Control: Optimization, Estimation, and Control*. Taylor and Francis.

Canepa, E. S. and Claudel, C. G. (2012). Exact solutions to traffic density estimation problems involving the lighthill-whitham-richards traffic flow model using mixed integer programming. In *Intelligent Transportation Systems (ITSC), 15th International IEEE Conference on*, pages 832–839.

Cattivelli, F. S. and Sayed, A. H. (2010). Diffusion lms strategies for distributed estimation. *Signal Processing, IEEE Transactions on*, 58(3):1035–1048.

Cheng, W., Bo, Y., Lijun, L., and Hua, H. (2008). A modified particle swarm optimization-based human behavior modeling for emergency evacuation simulation system. In *Information and Automation, 2008, IEEE International Conference on*, pages 23–28.

Chiang, Y.-T., Huang, P.-Y., Chen, H.-W., and Chang, F. R. (2008). Estimation of 3-d transformation from 2-d observing image using dual quaternion. In *World Congress*, volume 17, pages 10445–10450.

Choi, W., Hamacher, H. W., and Tufekci, S. (1988). Modeling of building evacuation problems by network flows with side constraints. *European Journal of Operational Research*, 35(1):98–110.

Claudel, C. G. (2010). *Convex formulations of inverse modeling problems on systems modeled by Hamilton-Jacobi equations. Applications to traffic flow engineering.* PhD thesis, University of California, Berkeley.

Claudel, C. G. and Bayen, A. M. (2010a). Lax–hopf based incorporation of internal boundary conditions into hamilton–jacobi equation. part i: Theory. *Automatic Control, IEEE Transactions on*, 55(5):1142–1157.

Claudel, C. G. and Bayen, A. M. (2010b). Lax–hopf based incorporation of internal boundary conditions into hamilton-jacobi equation. part ii: Computational methods. *Automatic Control, IEEE Transactions on*, 55(5):1158–1174.

Clifford, W. K. (1873). Preliminary sketch of bi-quaternions. *Proc. London Math. Soc*, 4(381-395):157.

Conn, A. R., Gould, N. I., and Toint, P. L. (2000). *Trust region methods*. SIAM.

Cornuejols, G. and Tütüncü, R. (2006). *Optimization methods in finance*, volume 5. Cambridge University Press.

Crano, W. D. (2000). Milestones in the psychological analysis of social influence. *Group Dynamics: Theory, Research, and Practice*, 4(1):68.

Daganzo, C. F. (1995). Properties of link travel time functions under dynamic loads. *Transportation Research Part B: Methodological*, 29(2):95–98.

Dai, R., Dong, J., and Sharma, A. (2015). Distributed traffic control for reduced fuel consumption and travel time in transportation networks. In *Control Conference (ECC), 2015 European*, pages 2658–2663.

Dai, R. and Sun, C. (2015). Path planning of spatial rigid motion with constrained attitude. *Journal of Guidance, Control, and Dynamics*, pages 1–10.

Dolev, D., Zymnis, A., Boyd, S. P., Bickson, D., and Tock, Y. (2009). Distributed large scale network utility maximization. In *Information Theory, ISIT 2009. IEEE International Symposium on*, pages 829–833.

Dong, J., Houchin, A. J., Shafieirad, N., Lu, C., Hawkins, N. R., and Knickerbocker, S. (2015). Vissim calibration for urban freeways.

Everett III, H. (1963). Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations research*, 11(3):399–417.

Filipe, N., Kontitsis, M., and Tsiotras, P. (2015). Extended kalman filter for spacecraft pose estimation using dual quaternions. *Journal of Guidance, Control, and Dynamics*.

Filipe, N. and Tsiotras, P. (2013). Simultaneous position and attitude control without linear and angular velocity feedback using dual quaternions. In *2013 American Control Conference*, pages 4808–4813.

Filipe, N. and Tsiotras, P. (2014). Adaptive position and attitude-tracking controller for satellite proximity operations using dual quaternions. *Journal of Guidance, Control, and Dynamics*, 38(4):566–577.

Filippidis, L., Galea, E. R., Gwynne, S., and Lawrence, P. J. (2006). Representing the influence of signage on evacuation behavior within an evacuation model. *Journal of Fire Protection Engineering*, 16(1):37–73.

Filippoupolitis, A., Hey, L., Loukas, G., Gelenbe, E., and Timotheou, S. (2008). Emergency response simulation using wireless sensor networks. In *Proceedings of the 1st international conference on Ambient media and systems*, page 21. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Floyd, J. E., Hunt, S. P., Williams, F. W., and Tatem, P. A. (2005). A network fire model for the simulation of fire growth and smoke spread in multiple compartments with complex ventilation. *Journal of fire protection engineering*, 15(3):199–229.

Funda, J. and Paul, R. P. (1990). A computational analysis of screw transformations in robotics. *Robotics and Automation, IEEE Transactions on*, 6(3):348–356.

Funda, J., Taylor, R. H., and Paul, R. P. (1990). On homogeneous transforms, quaternions, and computational efficiency. *Robotics and Automation, IEEE Transactions on*, 6(3):382–388.

Gabay, D. and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40.

Gaing, Z.-L. (2004). A particle swarm optimization approach for optimum design of pid controller in avr system. *IEEE transactions on energy conversion*, 19(2):384–391.

Garin, F. and Schenato, L. (2010). A survey on distributed estimation and control applications using linear consensus algorithms. In *Networked Control Systems*, pages 75–107.

Gawad, A. F. A. and Ghulman, H. A. (2015). Prediction of smoke propagation in a big multi-story building using fire dynamics simulator (fds). *American Journal of Energy Engineering*, 3(4-1):23–41.

Gershenfeld, N. A. (1999). *The nature of mathematical modeling*. Cambridge university press.

Goddard, J. S. and Abidi, M. A. (1998). Pose and motion estimation using dual quaternion-based extended kalman filtering. In *Photonics West'98 Electronic Imaging*, pages 189–200. International Society for Optics and Photonics.

Goldbergt, D. E. (1992). Control system optimization using genetic algorithms. *Journal of Guidance, Control, and Dynamics*, 15(3).

Gorbett, G. E., CFEI, C., and IAAI-CFI, M. (2008). Computer fire models for fire investigation and reconstruction. In *International Symposium on Fire Investigation and Technology*, pages 23–34.

Gorbil, G., Filippoupolitis, A., and Gelenbe, E. (2011). Intelligent navigation systems for building evacuation. In *Computer and Information Sciences II*, pages 339–345.

Grant, M. and Boyd, S. (2008). Cvx: Matlab software for disciplined convex programming (web page and software). *URL http://stanford. edu/boyd/cvx*.

Greenshields, B., Channing, W., Miller, H., et al. (1935). A study of traffic capacity. In *Highway research board proceedings*, volume 1935. National Research Council (USA), Highway Research Board.

Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on systems, man, and cybernetics*, 16(1):122–128.

Groves, P. D. (2013, pp. 163). *Principles of GNSS, inertial, and multisensor integrated navigation systems*.

Guler, S. I., Menendez, M., and Meier, L. (2014). Using connected vehicle technology to improve the efficiency of intersections. *Transportation Research Part C: Emerging Technologies*, 46:121–131.

Hadj-Salem, H., Blosseville, J., and Papageorgiou, M. (1990). Alinea: a local feedback control law for on-ramp metering; a real-life study. In *Road Traffic Control, 1990., Third International Conference on*, pages 194–198. IET.

Hamacher, H. W. and Tjandra, S. A. (2001). *Mathematical modelling of evacuation problems: A state of art*. Fraunhofer-Institut für Techno-und Wirtschaftsmathematik, Fraunhofer (ITWM).

Hamilton, W. R. (1844). Ii. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 25(163):10–13.

Han, K., Liu, H., Gayah, V. V., Friesz, T. L., and Yao, T. (2016). A robust optimization approach for dynamic traffic signal control with emission considerations. *Transportation Research Part C: Emerging Technologies*, 70:3–26.

Hegyi, A., De Schutter, B., and Hellendoorn, H. (2005). Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C: Emerging Technologies*, 13(3):185–209.

Hegyi, A., Hoogendoorn, S., Schreuder, M., Stoelhorst, H., and Viti, F. (2008). Specialist: A dynamic speed limit control algorithm based on shock wave theory. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*, pages 827–832.

Jahangiri, A., Afandizadeh, S., and Kalantari, N. (2011). The optimization of traffic signal timing for emergency evacuation using the simulated annealing algorithm. *Transport*, 26(2):133–140.

Jamshidnejad, A., Papamichail, I., Papageorgiou, M., and De Schutter, B. (2017). Sustainable model-predictive control in urban traffic networks: Efficient solution based on general smoothening methods. *IEEE Transactions on Control Systems Technology*.

Jelasity, M. (2011). Gossip. In *Self-organising software*, pages 139–162. Springer.

Jelasity, M., Montresor, A., and Babaoglu, O. (2005). Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems (TOCS)*, 23(3):219–252.

K. Holmstrom, A. G. and Edvall, M. M. (2005). User's guide for tomlab/miqq.

Kan, Y., Wang, Y., Papageorgiou, M., and Papamichail, I. (2016). Local ramp metering with distant downstream bottlenecks: A comparative study. *Transportation Research Part C: Emerging Technologies*, 62:149–170.

Kelly, F. P., Maulloo, A. K., and Tan, D. K. (1998). Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252.

Kisko, T. M., Francis, R., and Nobel, C. (1998). Evacnet4 users guide. *University of Florida*.

Komodakis, N., Paragios, N., and Tziritas, G. (2007). Mrf optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Komodakis, N., Paragios, N., and Tziritas, G. (2011). Mrf energy minimization and beyond via dual decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):531–552.

Li, L., Lv, Y., and Wang, F.-Y. (2016). Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 3(3):247–254.

Li, N. and Xu, Y. (2014). Evacuation modeling from the control perspective and corresponding sequential-based optimal evacuation guidance. *Control Systems Technology, IEEE Transactions on*, 22(3):1094–1102.

Li, Q., Fang, Z., Li, Q., and Zong, X. (2010). Multiobjective evacuation route assignment model based on genetic algorithm. In *Geoinformatics, 2010 18th International Conference on*, pages 1–5.

Li, Y., Canepa, E., and Claudel, C. (2014). Optimal control of scalar conservation laws

using linear/quadratic programming: Application to transportation networks. *Control of Network Systems, IEEE Transactions on*, 1(1):28–39.

Lighthill, M. J. and Whitham, G. B. (1955). On kinematic waves. ii. a theory of traffic flow on long crowded roads. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 229, pages 317–345.

Lin, Y.-H., Wang, H.-S., Chiang, Y.-T., and Chang, F.-R. (2010). Estimation of relative orientation using dual quaternion. In *2010 International Conference on System Science and Engineering*, pages 413–416.

Liu, S., Hellendoorn, H., and De Schutter, B. (2017). Model predictive control for freeway networks based on multi-class traffic flow and emission models. *IEEE Transactions on Intelligent Transportation Systems*, 18(2):306–320.

Liu, Y., Lai, X., and Chang, G.-L. (2006). Two-level integrated optimization system for planning of emergency evacuation. *Journal of transportation Engineering*, 132(10):800–807.

Low, S. H. and Lapsley, D. E. (1999). Optimization flow control. i. basic algorithm and convergence. *IEEE/ACM Transactions on networking*, 7(6):861–874.

Lu, C.-C., Mahmassani, H. S., and Zhou, X. (2008). A bi-criterion dynamic user equilibrium traffic assignment model and solution algorithm for evaluating dynamic road pricing strategies. *Transportation Research Part C: Emerging Technologies*, 16(4):371–389.

Lu, Q., George, B., and Shekhar, S. (2005). Capacity constrained routing algorithms for evacuation planning: A summary of results. In *Advances in spatial and temporal databases*, pages 291–307.

Lui, C., Fong, N., Lorente, S., Bejan, A., and Chow, W. (2015). Constructal design of evacuation from a three-dimensional living space. *Physica A: Statistical Mechanics and its Applications*, 422:47–57.

Luo, L., Ge, Y.-E., Zhang, F., and Ban, X. J. (2016). Real-time route diversion control in a model predictive control framework with multiple objectives: Traffic efficiency, emission reduction and fuel economy. *Transportation Research Part D: Transport and Environment*, 48:332–356.

Mamada, S., Makino, K., and Fujishige, S. (2004). Evacuation problems and dynamic network flows. In *SICE 2004 Annual Conference*, volume 1, pages 530–535.

Mazaré, P.-E., Dehwah, A. H., Claudel, C. G., and Bayen, A. M. (2011). Analytical and grid-free solutions to the lighthill–whitham–richards traffic flow model. *Transportation Research Part B: Methodological*, 45(10):1727–1748.

Mehrotra, S. (1992). On the implementation of a primal-dual interior point method. *SIAM Journal on optimization*, 2(4):575–601.

Mesbahi, M. and Egerstedt, M. (2010, Chaps. 8). *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press.

Mesmer, B. L. and Bloebaum, C. L. (2014). Incorporation of decision, game, and bayesian game theory in an emergency evacuation exit decision model. *Fire Safety Journal*, 67:121–134.

Messner, A. and Papageorgiou, M. (1990). Metanet: A macroscopic simulation program for motorway networks. *Traffic Engineering & Control*, 31(8-9):466–470.

Mirchandani, P. and Head, L. (2001). A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432.

Miyamoto, H. and Sasaki, S. (1997). Simulating lava flows by an improved cellular automata method. *Computers & Geosciences*, 23(3):283–292.

Morris, C. C. and Stark, R. M. (2015). *Finite Mathematics: Models and Applications*. John Wiley & Sons.

Murty, K. G. (1983). Linear programming.

Nedic, A. and Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61.

Nielsen, M. A. (2015). Neural networks and deep learning.

Ntziachristos, L., Samaras, Z., Eggleston, S., Gorissen, N., Hassel, D., Hickman, A., et al. (2000). Copert iii. *Computer Programme to calculate emissions from road transport, methodology and emission factors (version 2.1), European Energy Agency (EEA), Copenhagen*.

Olfati-Saber, R. (2009). Kalman-consensus filter: Optimality, stability, and performance. In *Decision and Control, held jointly with the 2009 28th Chinese Control Conference. Proceedings of the 48th IEEE Conference on*, pages 7036–7042.

Olsson, T., Bengtsson, J., Robertsson, A., and Johansson, R. (2003). Visual position tracking using dual quaternions with hand-eye motion constraints. In *Robotics and Automation, IEEE International Conference on*, volume 3, pages 3491–3496.

Papageorgiou, M., Hadj-Salem, H., and Blosseville, J.-M. (1991). Alinea: A local feedback control law for on-ramp metering. *Transportation Research Record*, (1320).

Papageorgiou, M., Hadj-Salem, H., and Middelham, F. (1997). Alinea local ramp metering: Summary of field results. *Transportation Research Record: Journal of the Transportation Research Board*, (1603):90–98.

Parkinson, A. R., Balling, R., and Hedengren, J. D. (2013). Optimization methods for engineering design. *Brigham Young University*, 5.

Pasquale, C., Papamichail, I., Roncoli, C., Sacone, S., Siri, S., and Papageorgiou, M. (2015). Two-class freeway traffic regulation to reduce congestion and emissions via nonlinear optimal control. *Transportation Research Part C: Emerging Technologies*, 55:85–99.

Pelechano, N. and Malkawi, A. (2008). Evacuation simulation models: Challenges in modeling high rise building evacuation with cellular automata approaches. *Automation in construction*, 17(4):377–385.

Peterson, C. K. and Paley, D. A. (2013). Distributed estimation for motion coordination in an unknown spatially varying flowfield. *Journal of Guidance, Control, and Dynamics*, 36(3):894–898.

Psakhie, S., Horie, Y., Ostermeyer, G., Korostelev, S. Y., Smolin, A. Y., Shilko, E., Dmitriev, A., Blatnik, S., Špegel, M., and Zavšek, S. (2001). Movable cellular automata method for simulating materials with mesostructure. *Theoretical and applied fracture mechanics*, 37(1):311–334.

Putha, R., Quadrifoglio, L., and Zechman, E. (2012). Comparing ant colony optimization and genetic algorithm approaches for solving traffic signal coordination under oversaturation conditions. *Computer-Aided Civil and Infrastructure Engineering*, 27(1):14–28.

Rangaiah, G. P. (2016). *Multi-objective optimization: techniques and applications in chemical engineering*, volume 5. World Scientific.

Richards, P. I. (1956). Shock waves on the highway. *Operations research*, 4(1):42–51.

Rush, A. M., Sontag, D., Collins, M., and Jaakkola, T. (2010). On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11. Association for Computational Linguistics.

Sagun, A., Anumba, C. J., and Bouchlaghem, D. (2014). Safety issues in building design to cope with extreme events: Case study of an evacuation process. *Journal of Architectural Engineering*, 20(3):05014004.

Savage, P. G. (2013). Blazing gyros: The evolution of strapdown inertial navigation technology for aircraft. *Journal of Guidance, Control, and Dynamics*, 36(3):637–655.

Schizas, I. D., Ribeiro, A., and Giannakis, G. B. (2008). Consensus in ad hoc wsns with noisy linkspart i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350–364.

Schütz, P., Tomasgard, A., and Ahmed, S. (2009). Supply chain design under uncertainty using sample average approximation and dual decomposition. *European Journal of Operational Research*, 199(2):409–419.

Shaw, J. W. and Noyce, D. A. (2014). Automated optimal balancing of traffic volume data for large access-controlled highway networks and freeway-to-freeway interchanges. In *Transportation Research Board 93rd Annual Meeting*, number 14-3565.

Shklovski, I., Palen, L., and Sutton, J. (2008). Finding community through information and communication technology in disaster response. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 127–136.

Shou-feng, L., Qin-ping, W., Wen, S., and Xi-min, L. (2012). Integrated simulation platform of vissim, excel vba, matlab. *Journal of Transportation Systems Engineering and Information Technology*, 12(4):43–48.

Sims, A. G. and Dobinson, K. W. (1980). The sydney coordinated adaptive traffic (scat) system philosophy and benefits. *Vehicular Technology, IEEE Transactions on*, 29(2):130–137.

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T. S., and Weiss, Y. (2012). Tightening lp relaxations for map using message passing. *arXiv preprint arXiv:1206.3288*.

Stengel, R. F. (2004, pp. 161-164). *Flight Dynamics*. Princeton University Press.

Sun, T. and Xin, M. (2015a). Multiple uav target tracking using consensus-based distributed high degree cubature information filter. In *AIAA Guidance, Navigation, and Control Conference*.

Sun, T. and Xin, M. (2015b). Multiple uav target tracking using consensus-based distributed high degree cubature information filter. In *AIAA Guidance, Navigation, and Control Conference*.

Tabirca, T., Brown, K. N., and Sreenan, C. J. (2009). A dynamic model for fire emergency evacuation based on wireless sensor networks. In *International Symposium on Parallel and Distributed Computing*, pages 29–36.

Tan, Q., Huang, G. H., Wu, C., and Cai, Y. (2011). If-em: An interval-parameter fuzzy linear programming model for environment-oriented evacuation planning under uncertainty. *Journal of Advanced Transportation*, 45(4):286–303.

Terelius, H. (2010). Distributed multi-agent optimization via dual decomposition. Master's thesis, Royal Institute of Technology, Stockholm, Sweden.

Tettamanti, T. and Varga, I. (2012). Development of road traffic control by using integrated vissim-matlab simulation environment. *Periodica Polytechnica. Civil Engineering*, 56(1):43.

Tjandra, S. A. (2003). Dynamic network optimization with application to the evacuation problem. *Doctoral Thesis, Technische Universitt Kaiserslautern*.

Tsitsiklis, J., Bertsekas, D., and Athans, M. (1986). Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812.

Tsitsiklis, J. N. (1984). Problems in decentralized decision making and computation. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR INFORMATION AND DECISION SYSTEMS.

VISION, P. (2014). Ptv vissim 7 user manual.

Wang, J. and Wilson, W. J. (1992). 3d relative position and orientation estimation using kalman filter for robot control. *Robotics and Automation*, pages 2638–2645.

Wang, J.-Y., Liang, H.-Z., Sun, Z.-W., Wu, S.-N., and Zhang, S.-J. (2013). Relative motion coupled control based on dual quaternion. *Aerospace Science and Technology*, 25(1):102–113.

Wang, P., Luh, P. B., Chang, S.-C., and Marsh, K. L. (2009). Efficient optimization of building emergency evacuation considering social bond of evacuees. In *Automation Science and Engineering, IEEE International Conference on*, pages 250–255.

Wei, E., Ozdaglar, A., and Jadbabaie, A. (2010). A distributed newton method for network utility maximization. In *Decision and Control, 49th IEEE Conference on*.

Wei, E., Ozdaglar, A., and Jadbabaie, A. (2013a). A distributed newton method for network utility maximization–i: Algorithm. *IEEE Transactions on Automatic Control*, 58(9):2162–2175.

Wei, E., Ozdaglar, A., and Jadbabaie, A. (2013b). A distributed newton method for network utility maximizationpart ii: Convergence. *IEEE Transactions on Automatic Control*, 58(9):2176–2188.

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Reviews of modern physics*, 55(3):601.

Wu, Y., Hu, X., Hu, D., Li, T., and Lian, J. (2005). Strapdown inertial navigation system algorithms based on dual quaternions. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(1):110–132.

Xiao, L. and Boyd, S. (2004). Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78.

Yao, T., Mandala, S. R., and Do Chung, B. (2009). Evacuation transportation planning under uncertainty: a robust optimization approach. *Networks and Spatial Economics*, 9(2):171–189.

Yue Zu, R. D. and Dong, J. (2016). Convex optimization for energy-efficient traffic control. In *IEEE Conference on Decision and control*.

Zargham, M., Ribeiro, A., Ozdaglar, A., and Jadbabaie, A. (2014). Accelerated dual descent for network flow optimization. *IEEE Transactions on Automatic Control*, 59(4):905–920.

Zarghamy, M., Ribeiroy, A., and Jadbabaiey, A. (2013). Accelerated dual descent for constrained convex network flow optimization. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 1037–1042. IEEE.

Zegeye, S. K. (2011). *Model-based traffic control for sustainable mobility*. TU Delft, Delft University of Technology.

Zenios, S. A. (2002). *Financial optimization*. Cambridge university press.

Zhao, D., Yang, L., and Li, J. (2008). Occupants behavior of going with the crowd based on cellular automata occupant evacuation model. *Physica A: Statistical Mechanics and its Applications*, 387(14):3708–3718.

Zu, Y. and Dai, R. (2017). Distributed path planning for building evacuation guidance. *Cyber-Physical Systems*, pages 1–21.

Zu, Y., Dai, R., and Dong, J. (2016). Convex optimization for energy-efficient traffic control. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 6759–6764. IEEE.

Zu, Y., Lee, U., and Dai, R. (2014a). Distributed motion estimation of space objects using dual quaternions. In *AIAA/AAS Astrodynamics Specialist Conference*, number 2014-4296, pages 4–7.

Zu, Y., Sun, C., and Dai, R. (2014b). Distributed estimation for spatial rigid motion based on dual quaternions. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 334–339. IEEE.

APPENDIX A. PROOF OF THEOREM 3.3.1

In subgradient method, we develop the fowllowing result associated with Euclidean distance to optimal set $\mathbf{x}^*$,

$$\begin{aligned} ||\mathbf{x}^{q_m+1} - \mathbf{x}^*||_2^2 &= ||\mathbf{x}^{q_m} - \alpha^{q_m} g_s^{q_m} - \mathbf{x}^*||_2^2 \\ &= ||\mathbf{x}^{q_m} - \mathbf{x}^*||_2^2 - 2\alpha^{q_m} g_s^{q_m T}(\mathbf{x}^{q_m} - \mathbf{x}^*) + (\alpha^{q_m})^2 ||g_s^{q_m}||_2^2 \\ &\leq ||\mathbf{x}^{q_m} - \mathbf{x}^*||_2^2 - 2\alpha^{q_m}(J^{q_m} - J^*) + (\alpha^{q_m})^2 ||g_s^{q_m}||_2^2. \end{aligned} \tag{0.1}$$

According to updating step in subgradient method, we expand $\mathbf{x}^{q_m+1} = \mathbf{x}^{q_m} - \alpha^{q_m} g_s^{q_m}$ in the first line in (0.1). Then the third inequality hold due to the definition of subgradient $g_s(\mathbf{x}^q)$ at $\mathbf{x}^q$. Recursively applying the result of (0.1) until $q = 0$ gives

$$\begin{aligned} ||\mathbf{x}^{q_m+1} - \mathbf{x}^*||_2^2 &\leq ||\mathbf{x}^{q_m} - \mathbf{x}^*||_2^2 - 2\alpha^{q_m}(J^{q_m} - J^*) + (\alpha^{q_m})^2 ||g_s^{q_m}||_2^2 \\ &\leq \cdots \\ &\leq ||\mathbf{x}^0 - \mathbf{x}^*||_2^2 - 2\sum_{q=0}^{q_m} \alpha^q (J^q - J^*) + \sum_{q=0}^{q_m} (\alpha^q)^2 ||g_s(x^q)||_2^2. \end{aligned} \tag{0.2}$$

As the Euclidean distance is equal to or greater than zero, the right side of (0.2) yields non-negativity, which implies the following inequalities where the second line holds by Assumption 3.3.2.

$$\begin{aligned} 2\sum_{q=0}^{q} \alpha^q (J^q - J^*) &\leq ||\mathbf{x}^0 - \mathbf{x}^*||_2^2 + \sum_{q=0}^{q_m} (\alpha^q)^2 ||g_s(\mathbf{x}^q)||_2^2 \\ &\leq U^2 + \sum_{q=0}^{q_m} (\alpha^q)^2 ||g_s(\mathbf{x}^q)||_2^2 \end{aligned} \tag{0.3}$$

Notice that the Euclidean distance achieves the minimum value at $J^i - J^* = J^{q_m}_{best} - J^*$, where i is the iteration index obtained by $i = \arg\min_j (J(x^j) - J^*), j = 0, ..., q_m$. Therefore, we have

$$2\sum_{q=0}^{q_m} \alpha^q (J^q - J^*) \geq 2(\sum_{q=0}^{q_m} \alpha^q)(J^{q_m}_{best} - J^*) \tag{0.4}$$

Combing the result of (0.4) with (0.3), we concludes the following inequalities, where the second inequality holds by Assumption 3.3.1.

$$\begin{aligned} J^{q_m}_{best} - J^* &\leq \frac{U^2 + \sum_{q=0}^{q_m} (\alpha^q)^2 ||g_s(\mathbf{x}^q)||_2^2}{2\sum_{q=0}^{q_m} \alpha^q} \\ &\leq \frac{U^2 + G^2 \sum_{q=0}^{q_m} (\alpha^q)^2}{2\sum_{q=0}^{q_m} \alpha^q} \end{aligned} \tag{0.5}$$

APPENDIX B. PROOF OF OBJECTIVE CONVERGENCE BY ADMM

Since $f_k(\mathbf{x}_k)$ is closed, proper and convex for $k = 1, ..., k$, $f(\mathbf{x})$ is also convex and differentiable. We take the partial derivative of $L_k(\mathbf{x}, \boldsymbol{\mu})$ with respect to $\mathbf{x}_k$ to provide

$$\partial L_k(\mathbf{x}_k^{q+1}, \mathbf{x}_{k'}^{q+1}, \mathbf{x}_{S'}^{q}, \boldsymbol{\mu}^q) = \partial f_k(\mathbf{x}_k^{q+1}) + \sum_{j=1}^{J} A_j^k \boldsymbol{\mu}_j^q + \rho \sum_{j=1}^{J} A_j^{kT} (\sum_{S'} A_j^{S'} \mathbf{x}_{S'}^q + \sum_{k'} A_j^{k'} \mathbf{x}_{k'}^{q+1} + A_j^k \mathbf{x}_k^{q+1} - \mathbf{b}_j) \tag{0.1}$$

where $k' \in \{1, \ldots, k-1\}$ and $S' \in \{k+1, \ldots, K\}$. As $\boldsymbol{\mu}_j$ is updated by $\boldsymbol{\mu}_j^{q+1} = \boldsymbol{\mu}_j^q + \rho \sum_{k=1}^{K} A_j^k \mathbf{x}_k^{q+1} - \mathbf{b}_j = \boldsymbol{\mu}_j^q + \rho \mathbf{r}_j^{q+1}$, we substitute $\boldsymbol{\mu}_j^q = \boldsymbol{\mu}_j^{q+1} - \rho \mathbf{r}_j^{q+1}$ into (0.1) as follows,

$$\begin{aligned} (0.1) &= \partial f_k(\mathbf{x}_k^{q+1}) + \sum_{j=1}^{J} A_j^{kT} \boldsymbol{\mu}_j^{q+1} \qquad (0.2) \\ &\quad -\rho \sum_{j=1}^{J} A_j^{kT} [\mathbf{r}_j^{q+1} - (\sum_{S'} A_j^{S'} \mathbf{x}_{S'}^q + \sum_{k'} A_j^{k'} \mathbf{x}_{k'}^{q+1} + A_j^k \mathbf{x}_k^{q+1} - \mathbf{b}_j)] \\ &= \partial f_k(\mathbf{x}_k^{q+1}) + \sum_{j=1}^{J} A_j^{kT} \boldsymbol{\mu}_j^{q+1} - \sum_{j=1}^{J} \sum_{S'} A_j^{kT} A_j^{S'} (\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^q) \qquad (0.3) \end{aligned}$$

Letting above expression equal to zero implies that $\mathbf{x}_k^{q+1}$ minimizes

$$f_k(\mathbf{x}_k) + \sum_{j=1}^{J} (\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'} (\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^q))^T A_j^k \mathbf{x}_k. \tag{0.4}$$

It turns out that

$$\begin{aligned} &f_k(\mathbf{x}_k^{q+1}) + \sum_{j=1}^{J} (\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'} (\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^q))^T A_j^k \mathbf{x}_k^{q+1} \\ &\leq f_k(\mathbf{x}_k^*) + \sum_{j=1}^{J} (\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'} (\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^q))^T A_j^k \mathbf{x}_k^* \qquad (0.5) \end{aligned}$$

It holds for $k = 1, ..., K$. Summing up those K inequalities gives the following inequality expressed as

$$f(\mathbf{x}) + \sum_{j=1}^{J}(\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'}(\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^{q}))^T \sum_{k=1}^{K} A_j^k \mathbf{x}_k^{q+1}$$
$$\leq f(\mathbf{x}^*) + \sum_{j=1}^{J}(\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'}(\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^{q}))^T \sum_{k=1}^{K} A_j^k \mathbf{x}_k^{*} \tag{0.6}$$

Notice $\sum_{k=1}^{K} A_j^k \mathbf{x}_k^* = \mathbf{b}_j$. Rearranging above inequality by , we have

$$f(\mathbf{x}^{q+1}) - f(\mathbf{x}^*) \leq -\sum_{j=1}^{J}(\boldsymbol{\mu}_j^{q+1} - \sum_{S'} A_j^{S'}(\mathbf{x}_{S'}^{q+1} - \mathbf{x}_{S'}^{q}))^T \mathbf{r}_j^{q+1} \tag{0.7}$$

Directly apply $\mathbf{r}_j^q \to 0$ as $q \to \infty$ as described in Boyd et al. (2011). We conclude

$$f(\mathbf{x}^{q+1}) - f(\mathbf{x}^*) \leq 0. \tag{0.8}$$

Moreover, Assumption 3.5.2 indicates

$$L(\mathbf{x}_1^*, \ldots, \mathbf{x}_K^*, \boldsymbol{\mu}^*) \leq L(\mathbf{x}_1^{q+1}, \ldots, \mathbf{x}_K^{q+1}, \boldsymbol{\mu}^*) \; at \; \rho = 0. \tag{0.9}$$

According to $\sum_{k=1}^{K} A_j^k \mathbf{x}_k^* = \mathbf{b}_j$, the left side of (0.9) is $f(\mathbf{x}^*)$. Based on $f(\mathbf{x}) = \sum_{k=1}^{K} f(\mathbf{x}_k)$, (0.9) is rewritten as

$$f(\mathbf{x}^*) \leq f(\mathbf{x}^{q+1}) + \boldsymbol{\mu}^{*T} \sum_{j=1}^{J} \mathbf{r}_j^{q+1}. \tag{0.10}$$

which implies $f(\mathbf{x}^*) - f(\mathbf{x}^{q+1}) \leq 0$ as $\mathbf{r}_j^q \to 0$. Combining (0.8), we have $f(\mathbf{x}^{q+1}) - f(\mathbf{x}^*) \to 0$ as $\mathbf{r}_j^q \to 0$, i.e. the objective convergence holds.

APPENDIX C. B-J/F EXPLICIT SOLUTIONS ASSOCIATED WITH INITIAL AND BOUNDARY CONDITIONS , AND FUNDAMENTAL DIAGRAM

C.1 Triangular-Model-Based B-J/F Explicit Solution

The relationship between Q and ρ is represented by a fundamental diagram $Q(\rho)$, which is established from empirical measurements. Triangular and parabolic shaped diagrams are two well-known curves representing the flow-density relationship. Triangular fundamental diagram is defined as,

$$Q(\rho) = \begin{cases} v_f \rho, & \text{if } 0 \le x \le \rho_c \\ w(\rho - \rho_j), & \text{if } \rho_c < x \le \rho_j \end{cases} \tag{1.1}$$

Given affine initial and boundary conditions described in (3.9)-(3.11), triangular-model-based solutions can be found in Mazaré et al. (2011), which are shown as follows.

For initial condition, it includes two cases, initially uncongested case when $0 \le \rho(0,x) \le \rho_c$, where ρ_c denotes the critical density,

$$N_{c_{ini}^k}(t,x) = \begin{cases} -\sum_{i=0}^{k-1} \rho(0,x_i)X + \rho(0,x)(v_f t + x_k - x), \text{ if } \dfrac{x - x_{k+1}}{v_f} \le t \le \dfrac{x - x_k}{v_f} \\ -\sum_{i=0}^{k-1} \rho(0,x_i)X + \rho_c(v_f t + x_k - x), \text{ if } \dfrac{x - x_k}{v_f} \le t \le \dfrac{x - x_{k+1}}{w}, \end{cases} \tag{1.2}$$

and initially congested case when $\rho_c \le \rho(0,x) \le \rho_j$,

$$N_{c_{ini}^k}(t,x) = \begin{cases} -\sum_{i=0}^{k-1} \rho(0,x_i)X + \rho(0,x)(wt + x_k - x) - \rho_j wt \text{ if, } \dfrac{x - x_k}{w} \le t \le \dfrac{x - x_{k+1}}{w} \\ -\sum_{i=0}^{k} \rho(0,x_i)X + \rho_c(wt + x_{k+1} - x) - \rho_j wt \text{ if, } t \ge \dfrac{x - x_{k+1}}{w}. \end{cases} \tag{1.3}$$

For upstream boundary condition, corresponding explicit solution based on Lax-Hopf formula is

$$N_{c_{up}^n}(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q(t_i,\xi)T + Q(t_n,\xi)(t - \frac{x-\xi}{v} - t_n), \text{ if } t_{n+1} + \frac{x-\xi}{v} \le t \le t_{n+1} + \frac{x-\xi}{v} \\ \sum_{i=0}^{n} Q(t_i,\xi)T + \rho_c v_f (t - \frac{x-\xi}{v} - t_{n+1}), \text{ if } t \ge t_{n+1} + \frac{x-\xi}{v}. \end{cases} \tag{1.4}$$

For downstream boundary condition, the corresponding explicit solution based on Lax-Hopf formula is

$$N_{c_{down}^n}(t,x) = \begin{cases} \sum_{i=0}^{n-1} Q(t_i,\chi)T + Q(t_n,\chi)(t - \frac{x-\chi}{w} - t_n) - \sum_{j=0}^{k_m} \rho(t,x_j)X - \rho_j(x-\chi), \\ \qquad \text{if } t_n + \frac{x-\chi}{w} \le t \le t_{n+1} + \frac{x-\chi}{w} \\ \sum_{i=0}^{n} Q(t_i,\chi)T + \rho_c v_f (t - \frac{x-\chi}{v} - t_{n+1}) - \sum_{j=0}^{k_m} \rho(t,x_j)X, \\ \qquad \text{if } t \ge t_{n+1} + \frac{x-\chi}{w} \end{cases} \tag{1.5}$$

Work in Canepa and Claudel (2012) verified the linearity and concavity associated with initial and boundary conditions.

C.2 Greenshields-Model-Based B-J/F Explicit Solution

In order to formulate the fuel-efficient traffic control problem as a convex optimization problem to improve computational efficiency, Greenshields model is employed which is defined as

$$Q(\rho) = -\frac{v_f}{\rho_j}\rho^2 + v_f\rho, \ \rho \in [0, \rho_j]. \tag{2.6}$$

Comparison between the two types of diagram in this specific application is described in section V.B. In the following, the focus is to find the exact solution based on the Greenshields model.

We first substitute $Q(\rho)$ in (3.13) by (2.6). Since $Q(\rho)-\frac{x-x_k}{t}\rho$ is concave, the supremum can be found by satisfying the first order necessary condition. Transformation of $R(\frac{x-x_k}{t})$ is expressed as

$$R(\frac{x-x_k}{t}) = \frac{v_f}{4}\rho_j + \frac{(x-x_k)^2\rho_j}{4v_f t^2} - \frac{x-x_k}{2t}\rho_j. \tag{2.7}$$

Based on the solutions to Moskowitz function provided in Mazaré et al. (2011), Q and R are replaced by (2.6) and (2.7), respectively. The B-J/F explicit solutions are obtained as follows. For initial condition, it includes two cases, initially uncongested case when $0 \le \rho(0,x) \le \rho_c$, where ρ_c denotes the critical density,

$$N_{c^k_{ini}}(t,x) = \begin{cases} (-\frac{v_f}{\rho_j}\rho(0,x_k)^2 + v_f\rho(0,x_k))t + c^k_{ini}(0,x), \\ \quad \text{if } \frac{x-x_{k+1}}{Q'(\rho_k)} \le t \le \frac{x-x_k}{Q'(\rho_k)} \\ \frac{v_f}{4}\rho_j t + \frac{(x-x_k)^2\rho_j}{4v_f t} - \frac{x-x_k}{2}\rho_j + c^k_{ini}(0,x_k), \\ \quad \text{if } t \ge \frac{x-x_k}{Q'(\rho_k)} \end{cases} \tag{2.8}$$

and initially congested case when $\rho_c \le \rho(0,x) \le \rho_j$,

$$N_{c^k_{ini}}(t,x) = \begin{cases} (-\frac{v_f}{\rho_j}\rho(0,x_k)^2 + v_f\rho(0,x_k))t + c^k_{ini}(0,x), \\ \quad \text{if } \frac{x-x_k}{Q'(\rho_k)} \le t \le \frac{x-x_{k+1}}{Q'(\rho_k)} \\ \frac{v_f}{4}\rho_j t + \frac{(x-x_{k+1})^2\rho_j}{4v_f t} - \frac{x-x_{k+1}}{2}\rho_j + c^k_{ini}(0,x_{k+1}), \\ \quad \text{if } t \ge \frac{x-x_{k+1}}{Q'(\rho_k)} \end{cases} \tag{2.9}$$

where $Q'(\rho_k) = \frac{dQ(\rho)}{d\rho}|_{\rho=\rho(0,x_k)}$. For upstream boundary condition, corresponding explicit solution based on Lax-Hopf formula is

$$N_{c^n_{up}}(t,x) = \begin{cases} \frac{(v_f - \frac{x-\xi}{t-t_n})^2\rho_j}{4v_f}(t-t_n) + c^n_{up}(t_n,\xi), \text{ if } t_n \le t \le t_n + T_0(\rho_{up}) \\ -\rho_{up}(x-\xi) + c^n_{up}(t,x), \text{ if } t_n + T_0(\rho_{up}) \le t \le t_{n+1} + T_0(\rho_{up}) \\ (v_f - \frac{x-\xi}{t-t_{n+1}})^2\rho_j(t-t_{n+1})/4v_f + c^n_{up}(t_{n+1},\xi), \\ \quad \text{if } t \ge t_{n+1} + T_0(\rho_{up}). \end{cases} \tag{2.10}$$

For downstream boundary condition, the corresponding explicit solution based on Lax-Hopf formula is

$$N_{c^n_{down}}(t,x) = \begin{cases} \frac{(v_f - \frac{\chi - x}{t_n - t})^2 \rho_j}{4v_f}(t - t_n) + c^n_{down}(t_n, \chi), \text{ if } t_n \le t \le t_n + T_0(\rho_{down}) \\ \rho_{down}(\chi - x) + c^n_{down}(t,x), \\ \quad \text{if } t_n + T_0(\rho_{down}) \le t \le t_{n+1} + T_0(\rho_{down}) \\ (v_f - \frac{\chi - x}{t_{n+1} - t})^2 \rho_j (t - t_{n+1})/4v_f + c^n_{down}(t_{n+1}, \chi), \\ \quad \text{if } t \ge t_{n+1} + T_0(\rho_{down}) \end{cases} \tag{2.11}$$

where $T_0(\rho_{up}) = \frac{x-\xi}{Q'(\rho_{up})}$, $T_0(\rho_{down}) = \frac{x-\chi}{Q'(\rho_{down})}$, $\rho_{up} = \min\{\rho \in [0, \rho_j] | Q(\rho) = Q(t,\xi)\}$, and $\rho_{down} = \max\{\rho \in [0, \rho_j] | Q(\rho) = Q(t,\chi)\}$.